AeroHydro

SurfaceWorks 8.5

Version 8.5

June, 2013

Portions under U.S. Pats. 5,581,672; 5,627,949; 5,856,828; Australia Pats. 659,919; 672,944.

AeroHydro, Inc.
PO Box 684 / 54 Herrick Rd.
Southwest Harbor, Maine 04679 USA
voice 207-244-4100
fax 877-734-6672
support@aerohydro.com
www.aerohydro.com

This manual was produced using Doc-To-Help®, by ComponentOne, Inc.

Contents

Getting Started

This Manual

This manual contains the introductory material necessary to get your SurfaceWorks program up and running. It consists of the Table of Contents, "Getting Started", the Glossary and Index from the full manual. The full manual may be found on your install CD or may be downloaded from the web.

What is SurfaceWorks?

Welcome to AeroHydro's family of surface modelers. The basic package consists of a complete computer aided design (CAD) package with the ability to create freeform surface shapes that in other CAD packages are either difficult to create or not available. There are a number of options that you can add to this base package to increase the modeler's functionality.

SolidWorks Integration

Used from within SolidWorks, this package increases SolidWorks capabilities by linking it to AeroHydro's superior surface creation tools.

Marine

For Naval Architects and Designers, we include in this package Offsets, Ship Lines and Hydrostatics. We find that the combination of our base modeler and these three tools is powerful enough to design most of our models.

Advanced Marine

This package adds Hydro to the Marine package. If you require calculation of flotation and righting moments of an elongated floating body, this is the package for you.

System Requirements

Minimum configuration:

Hardware environment - target platforms
Pentium (or higher) PC
128 MB RAM (minimum recommended)
35 MB free disk space (minimum)
Color Display
Mouse with center wheel (preferred)
Software environment - target platforms
Windows 2000© to Win8©

For SolidWorks Integration option
SolidWorks 2004 or higher

Installing SurfaceWorks

1 Start Windows.

2 Put the SurfaceWorks distribution CD in your CD drive.

3 If the install program does not start automatically, go to **Start>Settings>Control Panel** and double-click **Add/Remove Programs**.

3a Click **Add New Programs**.

3b Click the **CD or Floppy** button next to **To add a program from CD-ROM or floppy disc** and follow directions.

3c If you have to browse, look on the CD for **Setup.exe**.

3d Follow the remaining directions.

4 Follow the directions in the install program.

Built-in Licensing Software

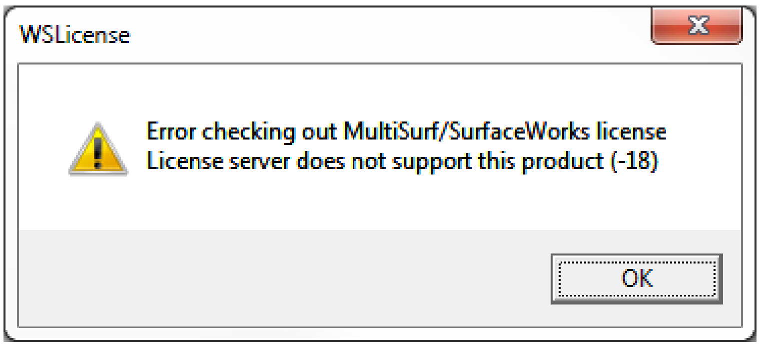
SurfaceWorks has a built in software licensing system to protect you and AeroHydro from unauthorized use of SurfaceWorks We have chosen this licensing system for its ease of use and the elimination of problems associated with hardware locking. Authorization is only an e-mail or phone call away.

Authorizing Your SurfaceWorks License

Your SurfaceWorks software comes preauthorized for a specified number of days (as found in your SurfaceWorks License dialog), the clock beginning when you first run SurfaceWorks After the expiration of the initial demo period, SurfaceWorks will not run until you have re-authorized your license. This means you can use the program immediately (or beginning at any time) for the specified period of consecutive days, but for continued use you must get your full authorization. You can apply for your full authorization at any time, even after the preauthorization period has expired. The basic process is: you give us your “Site Code,” then we issue you an “Authorization Key” (details below).

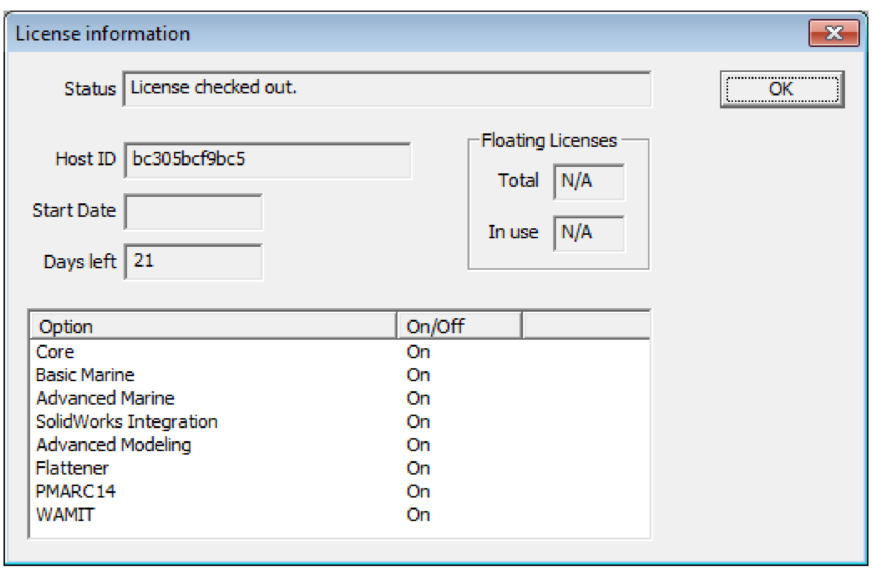
To Authorize Your License:

1a If your authorization or pre-authorization period has expired, the next time you start SurfaceWorks a message will appear on your screen:



Click <OK> to close the program. You may now continue with step **2** below.

1b If your pre-authorization period has not expired, choose **Help>Licensing**. The SurfaceWorks License dialog will appear.



2 **Look in the folder C:\Temp for the file 'hostid.txt'.** Email**, or fax, or your file to AeroHydro**. A license file will be returned with instructions of the proper location the file is to placed. (SurfaceWorks application folder)

Along with your hosted file, we also need:

- Program **name**, **Version #** and **Option Type** (i.e. SurfaceWorks , Advanced Marine, Flattener, ...). Program name and version can be found in **Help>About** SurfaceWorks or if this in not available, it can be found in the file **version.txt** found in the same folder as your program files.
- Your name

email: **support@aerohydro.com**

phone: **207-244-4100**

3 By authorizing your software you are agreeing to all terms of the software license agreement. You're now ready to roll with a fully authorized copy of SurfaceWorks !Welcome aboard.

Once authorized, you can review your current licensing status at any time by selecting **Help>Licensing**.

Note: Your SurfaceWorks license is site-specific to the particular folder on the particular machine in which you authorized it. If you want to move your copy to another machine, there is a specific procedure you need to follow; see "" on page 6. You may have your copy authorized in only one location at a time.

Network License Instructions

There are two install files. One for the Server and the other for the Clients. If the server needs to run MultiSurf/SurfaceWorks, both install files should be used.

Server:
32 bit: http://aerohydro.com/test/RLM_Server32.exe OR
64 it: http://aerohydro.com/test/RLM_Server64.exe

1. Download the appropriate installation file for the OS on your server.

2. Run the install file which will create a sub-folder "AeroHydro\License Info" in your \Users\AppData folder and copy some files. If a different location is desired, you will prompted in the process for any change you require.

3. The install creates an RLM Server and installs it as a service. The service is started after the process.

4. Created in the previous step is a file called 'hostid.txt' in your C:\Temp folder.

5. Send the "hostid.txt" file to AeroHydro so it can be used to generate your permanent license file.

6. Go to this address on your browser: http://<COMPUTER_NAME>:5054 to open the Server Admin Page. Bookmark this address.

7. A license file will be returned with further instructions for the server and clients.

8. That should be it for the server.

Client:

1. Download and Install MultiSurf.

2. MultiSurf should run in evaluation mode for 21 days.

3. Included with a new license file for the server, will be an additional file for the client. Please place this file in the MultiSurf/SurfaceWorks application folder.

4. Open ahi_client.lic in Notepad.

5. MultiSurf/SurfaceWorks should open as a client.

SurfaceWorks is now enabled for software licenses that work over a network. In this scheme, a user group purchases a shared license for X number of users.

Start SurfaceWorks and you should be OK

Supported Server Platforms

SurfaceWorks currently only supports the use of Windows 2000 through Windows 8 as the server platform.

License Transferal

Transferring Your License: From retired computer

If you require your license to be moved from an old computer, destined to be retired for MultiSurf use, simply remove MultiSurf and all of its program folders. It will now be necessary to install MultiSurf on the new computer and apply for a replacement license file.

Transferring Your License: Temporary Transfer for Travel

Sometimes you want to transfer a SurfaceWorks license to a computer that already has a MultiSurf installed for the purposes of traveling. The way we facilitate this move is through an additional feature, called Roaming, of the Network License. An network license, along with its associated extra cost would be required.

The directions to allow roaming are as follows:

Let's call the computer which is to roam the "laptop". The process involves editing the computer's Environment Variables. (EV) These can be found here: Control Panel\System and Security\System\Advanced System Settings\Environment Variables

1. Close all instances of SurfaceWorks
2. With the laptop connected to the local network, either wired or wirelessly, open the EV dialog.
3. The RLM_ROAM EV is located with the "System" EV's.
4. Edit RLM_ROAM and change the value to the number of days you wish to roam.
5. Open SurfaceWorks on the laptop and check the license through Help/Licensing. You should see a reference to one roaming license checked out.
6. The laptop can be disconnected from the local network with no loss of the SurfaceWorks license.

To check the license back in:

1. Connect the laptop to the local network.
2. Change the RLM_ROAM EV to -1.
3. Start SurfaceWorks on laptop and an error will appear.
4. Close SurfaceWorks
5. Change the RLM_ROAM EV to 0
6. Open SurfaceWorks on the laptop and check license to see that normal network license has returned.

Readme.txt

There is a file named Readme.txt installed into your application directory. It contains the most current information about your SurfaceWorks program — information so recent that it is not yet in our user manuals or the online help system.

You are offered the opportunity to view readme.txt at the end of the SurfaceWorks installation procedure. You can read the file any time you want by going to the Windows taskbar and choosing Start>Programs>AeroHydro> SurfaceWorks 4.x

Release Notes. You also can view and/or print the file from Windows Notepad or WordPad.

Help

The SurfaceWorks help system includes all the information and graphics that are in the User's Guide. You can access the Contents, Index and Search tabs by choosing Help>Help Topics. Additionally, you can use the context sensitive connections by highlighting a menu item (just point to the item, don't click) and pressing F1 or by pressing F1 while the focus is in a dialog box.

Technical Support

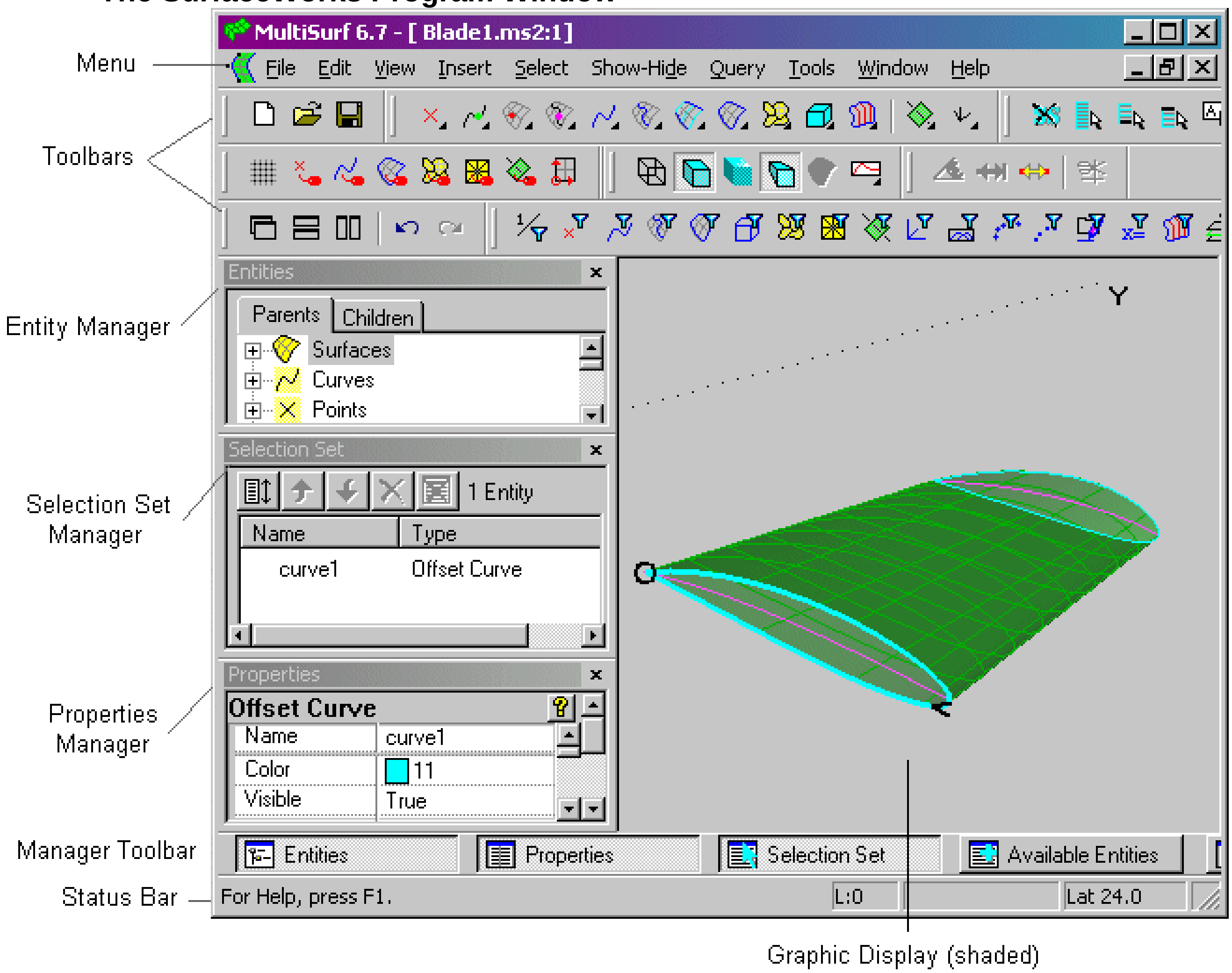
Your first line for technical support is the person or company from whom you purchased SurfaceWorks Beyond that, we at AeroHydro stand behind our product and will help you if you need further assistance. Please don't hesitate to get in touch! You can reach us at:

support@aerohydro.com
phone 207-244-4100; or 877-244-4141 (toll free in USA and Canada)

We'll always need to know your SurfaceWorks version and build numbers — you'll find them in **Help>About** SurfaceWorks

SurfaceWorks Terms

The SurfaceWorks Program Window



For details about the:

Menu options, refer to the topics in "Reference Guide to Functions".
Toolbars and toolbar buttons (identification), see "Toolbar Buttons" on page 40.
Entity Manager, see "The Entities Manager" on page 23.
Selection Set Manager, see "The Selection Set Manager" on page 28

Available Entities Manager, see "The Available Entities Manager" on page 29
Status Bar, see "The Status Bar" on page 39.
Graphic Display, see "Graphical Displays" on page 14.

Definitions

entity: A geometric entity. SurfaceWorks entities have relationships built into them.

parent: An entity upon which one or more other entities depends. A parent entity is named in the definition of each entity it supports. For example, four Curves are the parents for a Tangent Boundary Surface whose boundary they define; they are named in the surface's definition.

child: An entity built upon another entity. A child entity's existence depends on the existence of the previously-created entity. For example, a B-spline Curve is a child of each of the points which define it.

curve: A curve in 3D space. For more details, see "Basic Concepts - Parametric Curves".

In SurfaceWorks, rather than use such long terms as "point coincident with a curve" or "curve constrained to a surface," we use short names, like nicknames, for various classes of entities that are coincident with or constrained to another entity:

bead: A point coincident with (or constrained to) a curve — think of a bead on a necklace. For more details, see "Basic Concepts - Beads, Magnets, and Rings".

magnet: A point coincident with (or constrained to) a surface — think of a magnet on your refrigerator door. For more details, see "Basic Concepts - Beads, Magnets, and Rings".

snake: A curve coincident with (or constrained to) a surface — think of a snake sunning on the ground. For more details, see "Basic Concepts - Parametric Snakes".

ring: A point coincident with (or constrained to) a snake — think of a ringed pattern of scales on a snake. For more details, see "Basic Concepts - Beads, Magnets, and Rings".

Generic- vs. Specific-Entity-Type References

In the manuals and online help, you will see point, bead, magnet, and ring spelled sometimes all lowercase (e.g. point) and sometimes with an initial capital letter (e.g. Point). The incidences with initial capital (e.g. Projected Point) are the specific kind of entity. The all-lowercase incidences (e.g. point) are generic references, which include two or more specific kinds of entity (e.g. Point, Offset Point, Mirrored Point, and Projected Point).

Similarly, when we refer to a magnet (all lowercase), we are speaking of any kind of magnet, whereas when we refer to a Magnet (initial capital), we are speaking of a Magnet entity that is located on a surface by u, v values.

Note: References to points (all lowercase) generally mean all / any kind of point entity — that is, points, beads, magnets, and rings.

Model Windows

In SurfaceWorks you can have multiple models open and / or you can have numerous views of a single model open, each in a separate window.

Window Controls

The active window is the one on top. To activate another window, click in it or go to the Window menu and select the window you want from the list of open windows.

You control the size and position of windows using standard Windows controls: change size by dragging a corner or edge of the window; move the window by dragging its title bar. You can **Cascade, Tile Vertical**, or **Tile Horizontal** using the toolbar buttons or the Window menu. You can iconize (minimize) a window by clicking the  button in its upper right-hand corner.

Application, Window, and Model Settings

Some settings are globally based; you might think of them as being “owned by the application”. These settings are saved when you exit SurfaceWorks Application settings include:

- in Tools>Options
  all of the settings on the General, Display, and Entity tabs
  "Prompt after drag" and whether a snap is turned on (but not its value) on the Dragging tab
  Display Controls on the Performance tab
- on the View menu
  Status Bar on / off
  Toolbars on / off
  Display of Nametags can be made an Application setting by clicking the "Make Default" button in the View>Nametags dialog

Some settings are locally based; you might think of these as being “owned by the window”. None of these settings affect other open windows and all reset to their defaults when you close the window where they are set. For example, if you have a model open that has Grid turned on, and then close the model and re-open it, Grid will be reset to its default setting; off. If there is another model open at the same time, its Grid setting will be unaffected by these actions. Window settings include:

- in Tools>Options
  Accurate mode on the Sessions tab
- on the View menu
  Surface Curvature options
  Perspective / Orthogonal toggle
  view orientation
  Axes display (on / off)
  Grid display (on / off)
  Symmetry Images (on / off)
  Nametags
  Entity Orientation
- on the Select menu
  Filters

Current Layer Filter
Visible / Hidden / Both setting

- on the Edit toolbar and in Tools>Options, Dragging
  Orthogonal Dragging

Other settings relate specifically to a model, so you might think of them as being "owned by the model." These settings are saved with a model and include:

- in Tools>Options
  all of the settings on the Model and Model Units tabs
  Model settings on the Performance tab
  whether a snap is turned on (but not its value) on the Dragging tab
- on the Tools menu
  Layer names and layer on / off status

Graphical Displays

In SurfaceWorks, you create and edit entities in 3D space, that is, in the graphics displays, rather than on 2D sketches. You can have multiple work windows (graphics images) open at the same time, in which you might display a perspective view, one or more orthographic views, perhaps a zoomed-in view, etc. Edits you make in any of these images will be reflected in all the other images.

You can force the graphic display to redraw by pressing Enter while the pointer focus is in the window.

When you import an IGES file into SurfaceWorks, the model appears in the display mode you have chosen in your SurfaceWorks file.

To open additional work windows for the model (default image is shaded):

1 Choose **Window>New Model Window**.

2 To make the new image wireframe, click  or choose **View>Display>Wireframe**.

Local and global controls:

Local controls: In each of these windows, you can independently zoom, rotate, and pan; turn the axes and grid on / off; switch between perspective and orthographic view.

Global controls: Selecting an entity in one work window (or in the Entity Manager, or the Available Entities pane) selects the entity in all work windows (and the Entity Manager, and the Available Entities pane). SurfaceWorks notifies you of your selection by placing it in the Selection Set pane. Any changes made to the model in one work window will be updated in all work windows.

Shaded Display

The shaded display is a rendered, shaded, lighted image of the model. It has continuously variable transparency ranging from 0% (dark shading) to 100% (no shading; displays as wireframe). Additionally, you can opt to display surface mesh lines (the initial program default is to display them).

To set shaded display options:

Choose **Tools>Options>Display**. For details, see “Reference Guide to Functions - Display Options”.

To display a shaded image:

Click  or **View>Display>Shaded**.

Wireframe Display

The wireframe display shows the visible model as “wires.” It is a transparent image (no hidden lines).

To set wireframe display options:

Choose **Tools>Options>Display**. For details, see “Reference Guide to Functions - Display Options”.

To display a wireframe image:

Click  or **View>Display>Wireframe**.

Tips for Using the Wireframe Display

Wireframes tend to be easier to interpret in perspective view than in orthographic view. Also, if you find a particular wireframe image confusing, you generally can get clarifying visual cues by rotating the image, and watching how parts of the model move in relationship to other parts. (If the image is still confusing, you might try switching to a shaded display.)

Surface Curvature Display

The Surface Curvature display is a color-coded representation of normal, Gaussian, or mean curvature for the selected surface. The Surface Curvature display opens in a separate window. You can have multiple surface curvature windows open at the same time. The display opens in Rotate mode.

For a discussion about reading the Surface Curvature display, see “Reference Guide to Functions - Surface Curvature - Reading the Surface Curvature Display”. For definition and general discussion of surface curvatures and some applications of them, see “Editing Models - Fairing Surfaces”.

To open a surface curvature display:

1 **Select** a surface.

2 Click  or **View>Display>Surface Curvature**.

3 Choose the settings you want on the **Surface Curvature options** tab.

To set surface curvature options or change the color range of the display from a surface curvature window,

Choose **Tools>Options>Surface Curvature** or select **View Options** from the **right click menu**.

To open additional surface curvature displays:

1 Make a Surface Curvature window active.

2 Choose **Window>New Model Window**.

or

1 **Select** a surface (if not already selected).

2 Click  or **View>Display>Surface Curvature**.

Local controls:

Local controls: In each of these windows, you can independently set surface curvature options; zoom, rotate, and pan.

Curvature Profile Display

The curvature profile display is a graph of curvature vs. arc length for the selected curve (this will not work for Mirrored, Relative or Projected Curves or any snakes). This option is only available when a graphic window is open and active and a single qualifying curve is selected. The curvature profile opens a pane attached to the bottom of the active window.

To open a curvature profile display:

1 **Select** a curve of the allowable type.

2 Choose  or **View>Display>Curvature Profile.**

For a discussion and examples of reading the curvature profile display, see “Editing Models - Fairing Curves”.

3D Viewing

The 3D view can be controlled via mouse, keyboard, or menu. The following work in the graphic, Surface Curvature and Offsets displays.

Keys and Buttons Available to Control the View

+**left mouse button;** **Arrow keys;** **Press center mouse wheel or button, move mouse**	View>Modify>Rotate (Default increment = 10 degrees. Adjust increment in **Tools>Options>Performance> Degrees per keystroke**)
+**left mouse button;** **Ctrl+arrow keys;** **Ctrl+press center mouse wheel or button, move mouse**	View>Modify>Pan

+**left mouse button;** **PgUp / PgDn keys;** **Rotate center mouse wheel**	View>Modify>Zoom In / Out

Note: If you hold down Shift while pressing an arrow key, PgUp, or PgDn, the effect on viewing parameters is 1 / 10 that of the unshifted key. For example, if you are rotating and Degrees per keystroke (Tools>Options, Performance) is set to 10 degrees, the view orientation will change by 1-degree increments.

; **F**	View>Modify>Zoom to Fit
+**left mouse button;** **Shift+press center mouse wheel or button, move mouse**	View>Modify>Zoom to Area
; **F10**	View>Modify>Zoom to Selection (This acts like Zoom to Fit in Surface Curvature and Offsets displays)

Note: The results of zooming and panning are much more predictable in orthographic views. In fact, in perspective it is possible to zoom the viewpoint right through the entity, so the whole model is behind you, resulting in a blank screen!

; **F8**	View>Modify>Previous View displays the most recent view that was allowed to draw completely.
	View>Mechanical View Orientation>Front, Back, Left, Right, Top, Bottom, Home view orientations (the “standard” views)
	For Marine users: View>Marine View Orientation>Bow, Stern, Port, Starboard, Deck, Keel, Home view orientations (the "standard" Marine views
	View>Display>Perspective view. This is a toggle — when it is on, the view is perspective; when it is off, the view is orthographic.
A	Toggle View>Axes on and off. (This is not available in Surface Curvature and Offsets displays).
; **G**	Toggle View>Grid on and off (graphics displays only). Grid lines are visible only in orthographic views that are perpendicular to one or two of the coordinate axes. Heavy grid lines (every 10th) are labeled. The status line reports the size of the smallest grid divisions at the far right as “Grid # ” only when the grid lines are visible. If grid is on but the model is rotated this information does not appear. (This is not available in Surface Curvature and Offsets displays).

You can make key presses in rapid succession, without waiting for the program to finish updating the drawing.

Controlling the View

There are two options for rotation mode: Free tumble and Z axis vertical. Set the rotate method in **Tools>Options>General, Rotation Constraint.**

Free tumble mode

In Free tumble mode the model rotates around imaginary axes that are vertical and horizontal in relation to the screen so the model can appear to tumble when you rotate it.

Z axis vertical mode

In Z axis vertical mode, the model still appears to rotate, but because the Z axis is constrained to be vertical, i.e. tilt = 0, the model never appears to tumble. Note that this does not mean that positive Z always points up, but that the axis is always vertical in the plane of the screen.

Note: There are 2 caveats about the Z axis being vertical:

1 When you are in perspective view, the Z axis will sometimes be a little angled. This an artifact of how the perspective view is generated.

2 Left View, Right View and Top View all have non-zero tilt, and Home View can have tilt set to non-zero. If you go to a view that has non-zero tilt and then rotate, the model will immediately jump so that the Z axis becomes vertical in the plane of the screen. If you want to be able to use these views and rotate from them, you should use the Free Tumble Rotation Constraint. (**Marine users** have the additional option of being able to use the Marine views, all of which have tilt=0).

Viewing in steps

You can think of the view of the model as being recorded by a camera, which is “where your eye is”. Picture this camera residing on the surface of a sphere, pointing inward toward the center of the sphere which is the look point (Fig. 2). After you zoom, pan or rotate, the camera ends up in a different position around the model.

The camera position is specified by four variables:

- latitude — camera angle above or below the horizontal (X,Y) plane
- longitude — camera angle to the left or right from looking along the positive X-axis
- radius — distance from the center of the sphere (the camera’s look point) to the camera; this defines the size of the sphere
- tilt — camera tilt from vertical

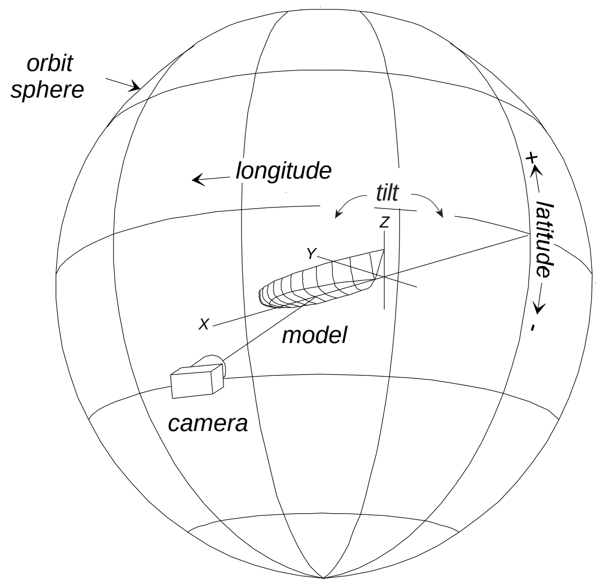


Fig. 2. Model and its axes, camera (your viewpoint), orbit sphere, latitude, longitude. The camera is at Lat -30, Lon 75, Tilt 0 in this view.

These four values are displayed on the status line. They are updated each time you rotate or zoom.

Rotate

Pressing the up and down arrow keys rotates the model around an imaginary screen horizontal axis that passes through the model's look point. In Z axis vertical mode, this changes only the latitude, in Free tumble mode both the latitude and longitude are free to change.

Pressing the left and right arrow keys:

in Z axis vertical mode rotates the model around an imaginary axis that is parallel to the Z axis and passes through the look point. This changes only the longitude.

in Free tumble mode rotates the model around an imaginary axis that is parallel to screen vertical and passes through the look point. Both longitude and latitude are free to change.

Pan

Holding down the <Ctrl> key while pressing any of the arrow keys pans the camera. This has the effect of moving the look point without changing latitude, longitude, radius or tilt.

Zoom

Pressing <PgUp> moves the camera inward; pressing <PgDn> moves the camera outward; these are changes in radius; the orbit sphere gets bigger and smaller

Tilt

Tilt cannot be controlled directly by any key. It is always 0 in Z axis vertical mode (except as noted abovein Z axis vertical mode). Tilt can be set for the Home view in Tools>Options, Model, Home View.

Smaller Motions

To move by smaller increments, hold down <Shift> while pressing the regular motion key(s).

Home

If you get lost, or want to get back to the beginning, click  or press <Home> or choose View>Mechanical Orientation>Home to restore the original view. If you want to change the Home view, go to Tools>Options, Model, Home View and set Latitude, Longitude and Tilt.

Degrees per Keystroke

In Z-axis vertical mode, pressing an arrow key changes the latitude or longitude the number of degrees specified in . If latitude is changed, longitude remains unchanged and vice versa.

In Free tumble mode, the model rotates around imaginary axes that are vertical and horizontal in relation to the plane of the screen. Because of this, the latitude and longitude changes are unpredictable. If the rotation happens to be perpendicular to one of the model axes, then the latitude and longitude will change the way it does in Z axis vertical mode, but only while the rotation remains perpendicular to that axis.

Dynamic viewing

You can also control the graphics view using the mouse. The mouse controls are equivalent to rotating, panning and zooming the camera using the arrow keys and <PgUp> / <PgDn>.

- You can select a method for controlling the view from the View>Modify menu and then use the left mouse button to move the model
- You can use the left mouse button to select the kind of motion desired from the View Modify toolbar and then use the left mouse button to move the model
- You can use the right mouse button to click in whitespace and then pick a mode from the menu and then use the left mouse button to move the model
- If you have a center mouse button you can use it to rotate or pan directly
- If you have a center mouse wheel, you can use it to rotate, pan or zoom directly.

Rotate

Place the pointer anywhere in the graphics view, press and hold down the center wheel (or button) of your mouse (if you have one) and move the mouse.

Alternatively, you can select View>Modify>Rotate or right click in white space and choose Rotate or just click  and then press and hold down the left mouse button in the graphics space while moving the mouse.

The cursor changes to  to indicate Rotation mode.

Pan

Press <Ctrl> then place the pointer anywhere in the graphics view and hold down the center wheel (or button) of your mouse (if you have one) and move the mouse.

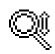
Alternatively, you can select View>Modify>Pan or right click in white space and choose Pan or just click  and then press and hold down the left mouse button in the graphics space while moving the mouse.

The cursor changes to  to indicate Pan mode.

Zoom In/Out

Move the top of the center mouse wheel away from you to zoom out or towards you to zoom in. If you use the wheel, the cursor shape does not change. The model zooms to the location of the cursor.

Alternatively, you can select View>Modify>Zoom In/Out or right click in white space and choose Zoom In/Out or just click  and then press and hold down the left mouse button in the graphics space while moving the mouse.

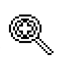
The cursor changes to  to indicate Zoom In\Out mode.

Move up to increase magnification, down to reduce.

Zoom to Area

Press <Shift> then place the pointer anywhere in the graphics view and hold down the center mouse wheel (or button). Drag a box around what you want to zoom in on or click and release to center the model on the click point.

Alternatively, you can select View>Modify>Zoom to Area or right click in white space and choose Zoom to Area or just click  and then press and hold down the left mouse button in the graphics space. Drag a box around what you want to zoom in on or click and release to center the model on the click point.

The cursor changes to  to indicate Zoom to Area mode.

Which Rotation constraint to use?

Which Rotation constraint should you use? Its all a matter of preference really. Z axis vertical is the viewing method that we originally used in SurfaceWorks. Some of us like it for boats and any large object because it gives us the feeling of walking around the model. For smaller objects, some of us like Free tumble mode because it gives us the idea of holding the object in our hands and turning it over any which way.

In Z axis vertical mode, you can pretty easily rotate with the mouse and get back to where you started. This is much harder with Free tumble mode. On the other hand, you can put the model in positions in Free tumble mode that cannot be reached in Z axis vertical mode.

Customizable Manager Dialogs

There are five customizable manager dialogs always available through a toolbar at the bottom of the workspace. These are the Entities Manager, Properties Manager, Selection Set Manager, Available Entities Manager and Errors Manager.

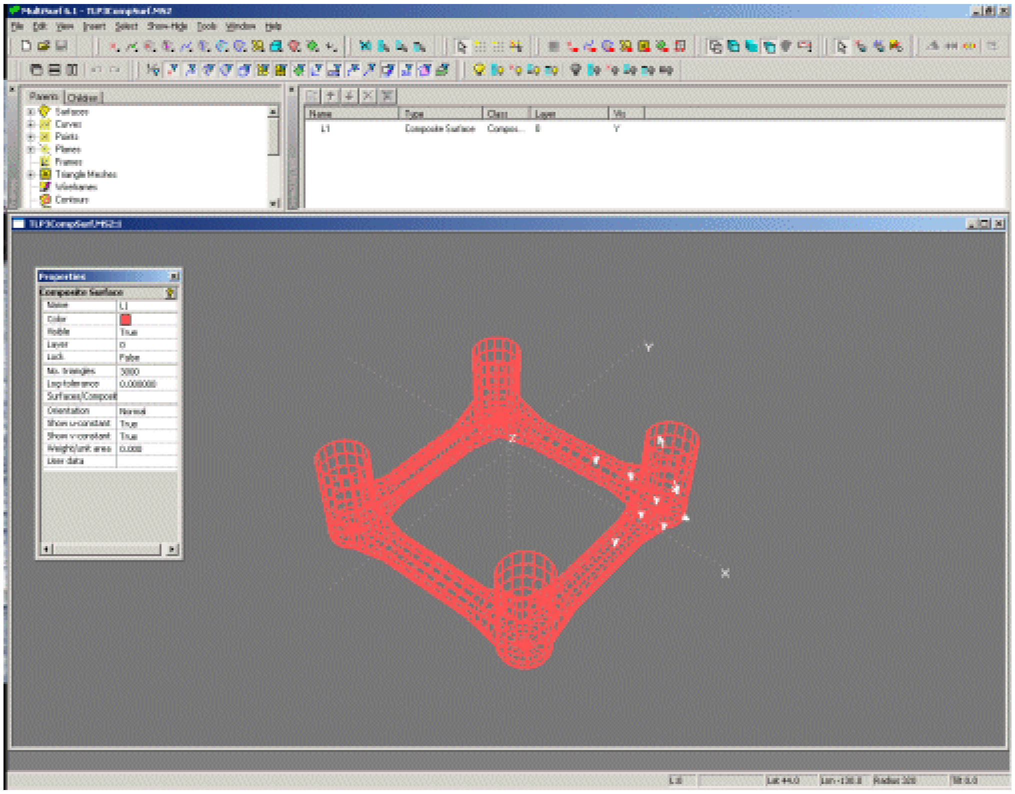
While SurfaceWorks continues to be a superb single screen tool, it has also been optimized for dual screen viewing. When the program is initially opened, three of these Manager views (Entities, Properties and Selection Set) are arranged from left to right in the wireframe window. These panes are independent, dockable, and able to be dragged to a second monitor.

In the image below, the Entities Manager and the Selection Set are arranged across the top. The wide view for the Selection Set gives the user the full benefit of all the information contained.

The Entities Manager is docked next to the Selection Set. It is not as long as one would like in this configuration, but is quite manageable for smaller models. A space for this manager on a second monitor would be best.

The Properties Manager is left to float on the screen. This way the full length of the dialog can be viewed. The dialog could be docked but that would cut in to available wireframe viewing area. If it is desirable to see a full unobstructed wireframe window, the Properties Manager could be temporarily closed and then quickly reopened by pressing the Properties button.

The Available Entities Manager is not open in this sample arrangement. We still suggest it remain closed with larger models, to improve performance. At least space is not an issue now, if two monitors are employed.



The Entities Manager

Entities Manager Functions

The Entities Manager presents information about the entities in your model in a fashion similar to the way Windows Explorer presents information about the files on your hard drive.

The Entities Manager allows you to:

- See a list of all entities.
- See dependencies between entities.
- See which entities have no children (typically they are involved in your work-in-progress).
- Select entities by name (even those which are hidden or on turned off layers).
- See which entities are selected.
- Select parents for an entity being inserted or edited.
- Rename an entity.
- See which entities are in error.

How the Entities Manager Works

- The root objects of the tree are the entity classes:

  Surfaces
  Curves
  Points
  Planes
  Frames
  Triangular Meshes
  Wireframes
  Contours
  Solids
  Composite Surfaces
  Relabels
  Graphs
  Knotlists
  Variables & Formulas
  Text Labels
  Solve Sets
  Entity Lists
  System

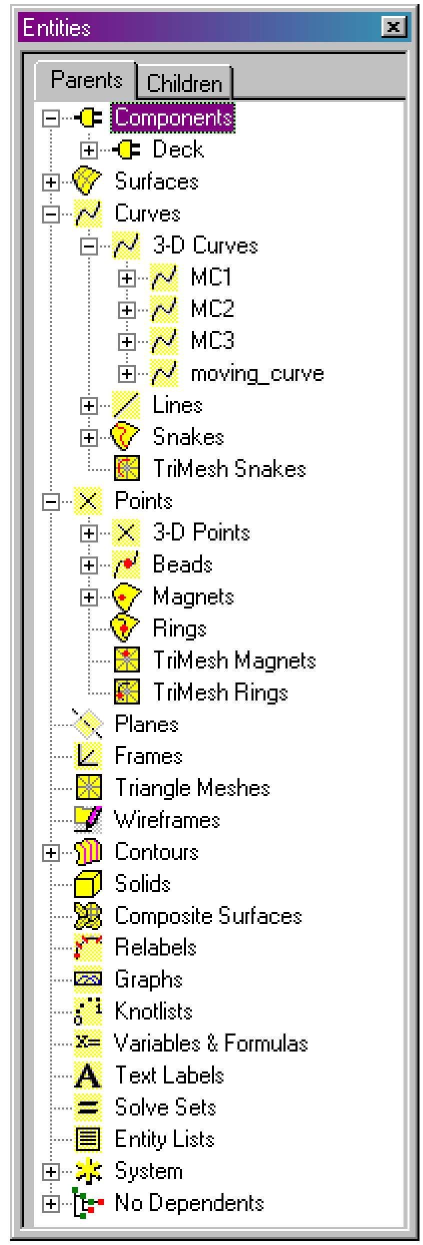
  and two non-entity classes:

  Components and No Dependents

  "Components" is a new heading which marks the listing of all components added to the model.

  "No Dependents" are points and curves that have no children. These entities also appear under the appropriate class, but are grouped here for easy access, since they typically are entities with which you are still working. You likely would be using them as direct or indirect parents for as-yet-uncreated surfaces.

  Beneath Curves and Points are sub-classes of these types.



- Expanding a class exposes the entities in that class. Entities appear in the tree by name and are sorted alphabetically. Placing the mouse pointer over the name generates a tooltip that displays the kind of entity plus the name. Expanding an entity exposes either the first generation of parents or children (depending on the state of the tab control). Note that these are usually not members of the class to which the top-level entity belongs.

- An entity may appear many times in the tree:

  Under its entity class
  As a parent or child of other entities (possibly many times)
  and / or
  As an entity with no dependents

- The two tabs are synchronized: The entities selected in one tab remain the selected entities in the other tab.

Mouse Actions and the Delete Key

In the Entity Manager:

- **Clicking** on an entity selects the entity, replacing whatever was previously in the Selection Set. The properties are now visible in the Property Manager.

- **Ctrl+clicking** an entity toggles its selection status without otherwise changing the Selection Set (e.g. adds an unselected entity to the Selection Set; removes a selected entity from the Selection Set).

- **Ctrl+Shift+clicking** an entity adds it to the Selection Set a second (or third...) time. You would use this, for instance, if you wanted to double a control point on a B-spline Curve.

- **SLOWLY double-clicking** on a name changes the highlight to an edit box, which allows you to rename the entity. If the resultant name is legal, the entity is renamed; otherwise, you get an error message and the edit is refused.

- **Right-clicking an entity** gets a shortcut menu.

- **Right-clicking in whitespace** (within the Surfer) gets a shortcut menu (different from the entity shortcut menu). One of the options on this menu is Clear Selection.

- Pressing **Delete** deletes the highlighted entity(ies) from the model (provided the entity(ies) has no children). You get a confirmation dialog (OK or Cancel).

Entities Manager Symbols

The Entities Manager uses the following symbols:

- Each entry is preceded by an icon indicating the general type of the entity or group: component, surface, curve, 3D point, plane, frame, triangular mesh, wireframe, contours, composite surface, relabel, entity list, No Dependents all appear as classes of entity, whereas: bead, magnet, ring, default object, snake, solid, knotlist, graph, variable & formula Text Label, Solve Set, all appear as parents or children of an entity.

- Error status is indicated by an overlaid on the entity causing the error. Entities in error only because a parent is in error are indicated by an overlaid white-on-red arrow, pointing down if the tree is in Parent mode, and up if the tree is in Child mode.

- The color of the entity's icon indicates its selection status: cyan (e.g.) = selected; yellow (e.g.) = not-selected.

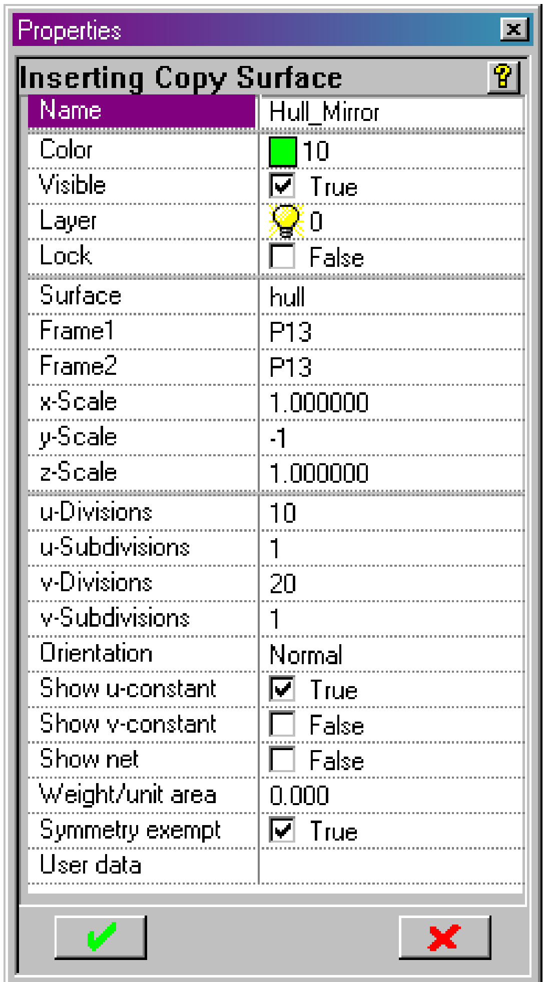
- Classes with entries and entities with parents / children have a ⊞ (expand) icon to their left, or if they are expanded, a ⊟ (collapse) icon.

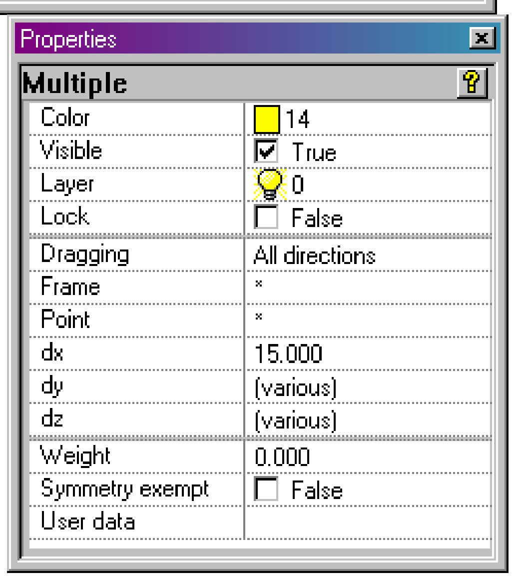
The Properties Manager

The example below shows how the Properties Manager would look for a Copy Surface. All fields are directly editable and in most cases the changes take place with no confirmation needed. The Properties Manager appears upon selection of the object.

If more than one object is selected a Multiple Edit Properties Manager appears. The properties that are available for multiple editing appear in the Multiple Edit Properties Manager.

The same Properties Manager is used for the Insertion of objects and editing. One difference between Insertion and Editing is that a confirmation would be needed to complete the Insertion process.



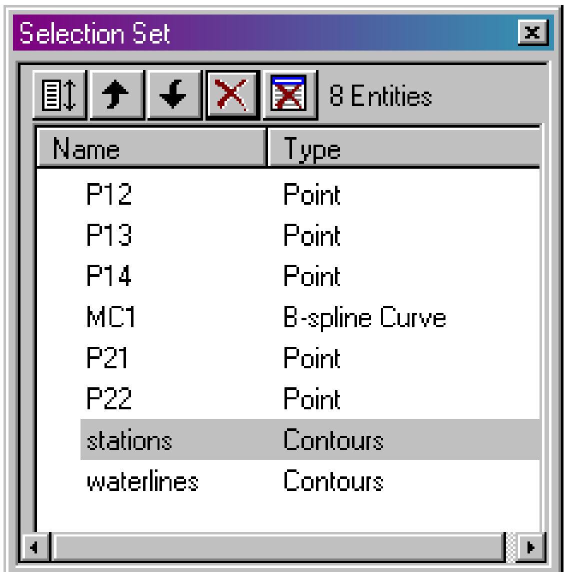


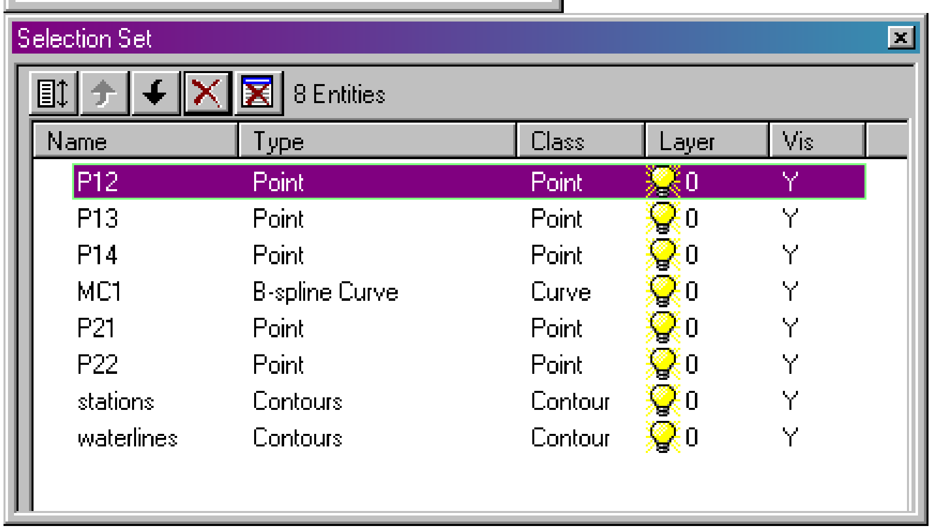
See also: Using the Properties Manager Effectively in Editing Models

The Selection Set and Available Entities Managers

The Selection Set Manager

The Selection Set dialog displays the contents of the Selection Set and allows you to change the order of the entities in it. The list has sortable columns for Name, Type, Class, Layer and Visibility. To increase or reduce the number of columns showing, use the horizontal scroll bar or the vertical split bar that separates the Selection Set Manager from the drawing space. To expand the pane vertically, click ▲ or drag its upper or lower split bar. To collapse the manager, click ▼ or drag its upper or lower split bar.





By default, the entities in the Selection Set manager are listed in the order in which they are selected.

When a selected entity is being edited, it remains shown in the Selection Set manager except when its parent entities are being specified. In this state, the selected parents are listed in the Selection Set manager, temporarily replacing the entity being edited.

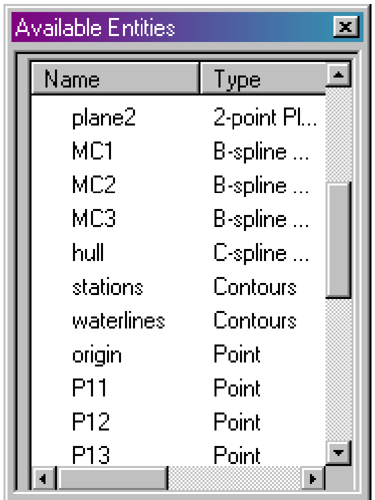
When specification of parents is finished, the entity being edited again is listed in the Selection Set manager.

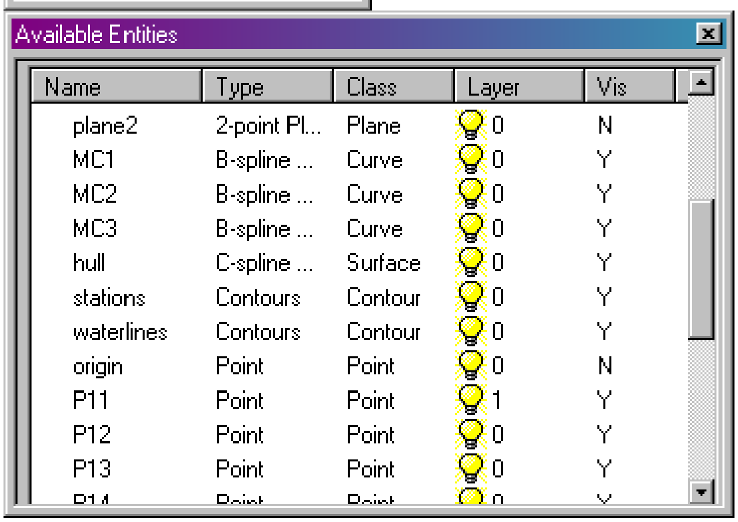
- To put one or more entities into the Selection Set, you must **select** them from the graphics displays, from the Entity Manager, or the Available Entities manager (for details about selecting, see "Selecting Entities" on page 31).
- Clicking sorts the list in the reverse order, moves the highlighted entity up the list, moves the highlighted entity down the list.
- You can remove an entity from the Selection Set by **highlighting it in the Selection Set list and clicking** , or by **Ctrl+clicking it in the Available Entities manager, or in the Entity Manager, or in a graphics display**.
- If there is a need to edit one object from a long list of selected objects, select the object and click to remove all but the selected object. The properties of the selected object now appear in the Property Manager for editing.
- You can delete an entity from the model by highlighting it and pressing the **Delete** key; then click OK or Cancel to the confirmation dialog.
- **Right-clicking** an entity or in whitespace gets (different) shortcut menus. For details, see "Shortcut Menus" on page 36.

The Available Entities Manager

The Available Entities manager displays the entities currently available for selection. These entities are the ones that can get through any filters which have been set or, if parents are being chosen for an entity, the entities that qualify as legal parents. The list has sortable columns for Name, Type, Class, Layer and Visibility. To increase or reduce the number of columns showing, use the horizontal scroll bar or the vertical split bar that separates the Available Entities Manager from the drawing space. To expand the pane, click or drag its upper or lower split bar. To collapse the pane, click or drag its upper or lower split bar.

The Available Entities Manager is closed, by default, in versions of SurfaceWorks since 5.0. With the improvements to the Entity Manager and the adoption of the Property Manager, there is less need to use this pane. It is our suggestion to leave it closed to give more room for the aforementioned panes and to optimize program performance.





- **Click**, **Ctrl+click**, and **Ctrl+shift+click** put individual and multiple entities into the Selection Set (reflected in the Selection Set manager, the Entity Manager, and any graphics displays).

- Pressing **Delete** deletes the highlighted entity from the model. You get a confirmation dialog (OK or Cancel).

- **Right-clicking** an entity or in whitespace gets you (different) shortcut menus. For details, see "Shortcut Menus" on page 36.

Sorting the List

The default sort is by Class. To sort the list according to a different column, click the column-heading button. Each column sorts in ascending or descending alphabetical order. To reverse the sort order (e.g. change from ascending to descending), click the column-heading button again.

Selecting Entities

In SurfaceWorks, the Selection Set is "owned by the document" and reflected in any graphics displays, the Entity Manager, and the Selection Set pane. You can select entities (i.e. put them in the Selection Set) from four windows: the graphics displays (directly), the Entity Manager or Available Entities pane (by name).

In Select Mode:

- **Clicking** on an entity that can get through the filters selects the entity, replacing whatever was previously in the Selection Set. (Any entity that exists in the model, including filtered, hidden or any other can be selected in the Entity Manager).

- **Ctrl+clicking** an entity toggles its selection status without otherwise changing the Selection Set (e.g. adds an entity to the Selection Set; removes a selected entity from the Selection Set).

- **Ctrl+Shift+clicking** an entity adds it to the Selection Set a second (or third...) time. You would use this, for instance, when making a closed-loop B-spline Curve that begins and ends at the same point.

Not in Select Mode:

- If you are not in Select Mode you can still select entities by **Clicking** on an entity in the Entity Manager or Available Entities Pane.

Selecting and Filters

If you try to select an entity and it "won't select," most likely it can't get through your current filter settings. Reset your filters and you'll be able to select the entity.

Note: If you have sound for Fail Filters on (**Tools>Options, General, Sounds**), your computer will beep (or chirp or whatever) at you when you try to select an entity and no entities in the target area can get through the filters.

Selecting in the Graphics Windows

You can use the graphics displays to select any visible (i.e. not hidden) entity, provided that kind of entity can get through any filters you may have set. To select entities in these graphic images, the pointer must be in Select Mode.

Putting the Pointer in Select Mode

There are several ways you can put the pointer in Select Mode:

- Choose **Select>Select Mode**.

- Press **Esc** when in another mode (e.g. Rotate, Zoom In/Out).

- **Right-click in whitespace in the graphics display** and choose **Select Mode** from the shortcut menu.

- **Click the current viewing mode button again** to turn it off

- Click .

The pointer shape lets you know what kind of entity you are pointing to, e.g.:   for a point. For a complete list see "Pointer Shapes" on page 45.

Selecting Entities One by One

To select a point:

Simply click or Ctrl+click on it (if you have nametags on, be sure to click on the point, not the nametag). When selected, it will be displayed larger in size.

To select a curve or a snake:

Click or Ctrl+click anywhere along its length. When selected, it will be displayed wider and may display orientation marks depending on settings.

To select a surface:

Click or Ctrl+click on its boundary or any of its u, v mesh lines in wireframe view, or anywhere on the surface in shaded view. When selected, its u, v lines will be displayed wider and its orientation marks will be visible (if turned on).

To select a plane:

Click or Ctrl+click any of the lines used to represent it.

Note: To select a hidden entity, you must use the Entities Manager or use Select By Name (or you can use Select>All if either Select>Both or Select>Hidden is turned on).

Selecting Multiple Entities by Dragging

Select multiple entities simultaneously by dragging a selection rectangle in the graphics window.

1. Set filters as needed.
2. Point <u>in whitespace</u> to one corner of the area you'd like to enclose, press the left mouse button, drag to the diagonally opposite corner of your selection area, and release the mouse button.

Note: There are 2 methods of selecting entities by dragging a window. Should you drag a window from right to left, only entities which are completely enclosed by the window will be selected. Dragging from right to left will select all objects enclosed or touched by the selection window.

Select Mode Has a Built-in Filter

The Select Mode pointer has a selection aid built right into it: it looks at all the entities in or across the target area and gives you the one of lowest dimensionality (points being of lowest dimensionality, then curves, wireframes, contours, frames and planes in a group and then surfaces). Thus when all the entity class filters are on, if you aim the pointer at a point entity, but a curve and surface are right there as well, SurfaceWorks selects the point. Or if you aim at a curve, but a surface is right there, too, SurfaceWorks selects the curve. The pointer shape indicates the class of the lowest-dimension entity (that is, the one it will pick if you click) — e.g.: for a point, for a curve.

Ambiguous Selections and the Which Entity Dialog

If the pointer is aimed at more than one entity of lowest dimensionality, the selection is ambiguous (SurfaceWorks is pretty smart, but it can't read your mind!) — the program will need you to tell it which of the entities you want. In ambiguous cases, the pointer displays as . When you click, you'll get the Which Entity dialog, which lists all the targeted entities that can get through the filters. The list is ordered by dimensionality, lowest first. Highlight the entity you want (click on it in the list), then click Select.

Clearing the Selection Set

To clear the Selection Set (leaving it empty), click , or choose **Select>Clear Selection Set,** or press **Ctrl+L**, or left click in whitespace in a graphics display.

Selecting from the Entities Manager

You can use the Entities Manager to select both visible and hidden entities as well as entities on turned off layers.

To select an entity:

Expand the tree if need be, then click (selects single entity), Ctrl+click (toggles entity status in Selection Set), or Ctrl+shift+click (to add an entity a second time) the entity or entities you want. When selected, an entity's icon is cyan (rather than yellow).

Selecting from the Available Entities Pane

You can use the Available Entities pane to select both visible and hidden entities, provided that kind of entity can get through any filters you may have set. Since the Available Entities pane only displays entities that can get through the filters (or legal parents, when parents are being selected), you will never run into the situation of trying to select an entity in this pane and finding that it won't select.

To select an entity:

Re-sort the list if need be, then click or Ctrl+click or Ctrl+shift+click the entity or entities you want. When selected, an entity's name is highlighted.

Errors Manager

The Errors Manager Dialog shows the details of any entities that are in error. For details see "What's Wrong" in the Reference chapter.

Filters

You can use filters to help control which entities are available for selection. When an entity is available for selection, it is listed in the Available Entities pane and if you click it in any of the text panes or a graphics window, it will become selected.

You select filters via the Select Menu or the toolbar buttons. Filter categories are:

Class (of entity)
Current Layer
Visible / Hidden / Both (menu only - no toolbar button)

When a filter is active (turned on), it allows you to select entities that match the filter.

The different kinds of filters are additive. For example: if for Class Filters you have only Points turned on AND if for the Visible / Hidden / Both option Visible is active, then only visible points will be available for selection.

Applying or changing filters does not affect entities already in the Selection Set.

Class Filters

You can activate or deactivate (in the **Select>Class Filters** dialog box or with **toolbar buttons**) the following entity class filters:

Invert Class Filters
Point
Curve
Snake
Surface
Composite Surface
Graphs
Variables and Formulas
Triangular Mesh
Plane
Frame
Relabel
Wireframe
Contours
Knotlists

When the points filter is on, point entities (3D points, beads, magnets, and rings) can be selected and points will be listed in the Available Entities pane.

Current Layer Filter

When the Current Layer filter (**Select>Current Layer Filter** or) is toggled on, you can select entities only from that layer. When this filter is off, you can select entities from any layer that is turned on. For usage, see "Example 2 — Current Layer Filter" on page 36.

Note: You set the current layer in **Tools>**Layers. This option also lets you actually turn layers on or off, in any combination. When a layer is off, all the entities on it are invisible and they are available for selection only using the Entity Manager.

Visible/Hidden/Both Filters

These filters consider visibility. They operate using a trio of mutually exclusive choices (via the **Select** menu). The default is Both. Clicking Visible or Hidden enables you to only select entities of that type.

Note: Hidden entities cannot be selected in the graphics displays, even if you have the hidden filter turned on. You'll need to use the Entity Manager, Available Entities pane or Select>All instead.

Layers

Layers allow you to group related portions of a model together for display and selection — every entity is on a layer; you can turn any layer on or off at will. The layering concept is similar to using transparent overlays in traditional drafting. Turning a layer off is like peeling off a transparent overlay — the entities still exist, but they aren't displayed or selectable (except in the Entity Manager) until you turn the layer back on. (Note that unlike traditional CAD programs, entities on layers that are turned off can be affected by changing a parent entity on a layer that is turned on.)

Newly created entities are put on the current layer.

Imported IGES geometry is put on the same layer (level) as it was in the imported file.

(For more information on layers, see "Reference Guide to Functions –Layers").

To set the current layer and toggle layers on/off:

Choose **Tools>Layers**.

Current Layer Filter

As previously mentioned in Filters, there is also a filter that restricts selection to the current layer only. This also is like working with a set of overlays — in this case, you can see all the overlays, but you can only erase or write on the topmost overlay, which we call the current layer. (Note that unlike traditional CAD programs, entities on layers other than the current layer can be affected by changing a parent entity that is on the current layer.)

To toggle the Current Layer filter:

Choose **Select>Current Layer Filter** or 

Example 1 — Turning Layers Off

Often, a model requires several levels of parents for the shape definition — including control points, master curves, and possibly a basis surface of which the surface used is a Trimmed Surface. In this kind of situation, one use of layers is to place all the construction entities on one layer, and the final surface(s) on another.

During the design of the basic shape, you'd keep both layers turned on. But once the basic design is finished, and you are adding entities attached to the basic shape, the construction entities are merely in the way. You could just hide these entities (see "Reference Guide to Functions - Hide"), but then any time you used one of the general Show commands (such as Show>Points), the construction points (which you don't want to see) would be displayed along with the new material you are working on. Using Tools>Layers, you can simply turn off the layer holding the construction entities, and they will not be displayed (even if you do a Show>All).

Example 2 — Current Layer Filter

When a model is fairly well-developed, typically there are a lot of entities on the screen, only a few of which you are working with. For example, when you're working on a headlamp dome, you generally would want to see the dome surface (to have a picture of the overall shape of the headlamp), but you wouldn't want the program to ask if you want to select the dome every time you click on an entity that happens to be in front of the dome.

To take advantage of the current layer filter:

1. Use Tools>Layers to set a new current layer when you are ready to create entities that will be attached to the dome.

2. Once you've made the basic attachment between, say, the dome and the housing and are constructing entities based on housing entities rather than dome entities, use Select>Current Layer Filter to turn on the Current Layer filter. Now, only entities on the current layer will be available for selection. This lets you quickly select and move entities in the housing without having to deal with the dome entities you are using for reference only.

3. If at any time you need to select an entity on another layer just select from the Entity Manager or you can turn the Current Layer filter off, pick the entity, then turn the filter back on.

Shortcut Menus

Right-clicking an entity or whitespace in various windows brings up an appropriate shortcut menu. This generally allows you to keep your mouse movements in a more compact area of the screen while you work.

Graphics Window Shortcut Menus

Right-clicking an entity in the graphics window displays the following shortcut menu:

Delete
Hide
Parent/Child
Host/Guest
Zoom to Fit
Zoom to Area
Zoom In/Out
Zoom to Selection
Rotate
Pan
Previous
View Options

Choosing Parent/Child brings up a dialog showing the first generation parents and children of the entity.

Right-clicking in whitespace in the graphics window when the pointer is in Select mode displays the following shortcut menu:

Show Selection Set (grayed if Selection Set is empty)
Hide Selection Set (grayed if Selection Set is empty)
Invert Selection Set
What's Wrong
Select Mode
Zoom to Fit
Zoom to Area
Zoom In/Out
Zoom to Selection (grayed if Selection Set is empty)
Rotate
Pan
Previous
View Options

Right-clicking in whitespace in the graphics window when the pointer is in any mode except Select (e.g. Zoom to Area, Rotate) displays the following shortcut menu:

Select Mode
Zoom to Fit
Zoom to Area
Zoom In/Out
Rotate
Pan
Previous
View Options

Right-clicking in the distance report box (The box generated by the Tools> Measure>Distance tool) in the graphics window when the pointer is in any mode displays the following shortcut menu:

Remove

Entity Manager Shortcut Menus

Right-clicking an entity in the Entity Manager displays the following shortcut menu:

Delete
Show or Hide (SurfaceWorks will put in the correct option)
Parent / Child
Host / Guest
What's Wrong (if entity is in error, grayed if not in error)

Right-clicking in whitespace in the Entity Manager displays the following shortcut menu:

Expand To Selected Entities (This option opens the tree sufficiently to ensure that every entity in the Selection Set is exposed.)
Collapse All
Clear Selection
Select All

Selection Set Pane Shortcut Menus

Right-clicking an entity in the Selection Set pane displays the following shortcut menu:

Delete
Hide or Show (SurfaceWorks will put in the correct option)
Parent / Child
What's Wrong (if applicable)

Right-clicking in whitespace in the Selection Set pane displays the following shortcut menu:

Clear Selection
Show Selection Set
Hide Selection Set

Available Entities Pane Shortcut Menus

Right-clicking an entity in the Available Entities pane displays the following shortcut menu:

Delete
Hide or Show (SurfaceWorks will put in the correct option)
Parent / Child
Host / Guest
What's Wrong (if applicable)

Right-clicking in whitespace in the Available Entities pane displays the following shortcut menu:

Clear Selection
Show Selection Set
Hide Selection Set

Surface Curvature window Shortcut Menus

Right-clicking anywhere in a Surface Curvature window displays the following shortcut menu:

Zoom to Fit
Zoom to Area
Zoom In / Out
Rotate
Pan
Previous
View Options

The Status Bar

The status bar at the bottom of the SurfaceWorks screen displays some very valuable information:

- (Left side of status bar) The identity (and for points, the location) of the last selected entity — to display this information, select the entity.
- While dragging, dynamic display of coordinates, and / or t, u, v values.
- Messages about what you are currently creating or editing. (Cultivate the habit of reading these! they can keep you sane while you navigate complex operations).
- Information about the highlighted menu item.
- (Right side of status bar) The number of the current layer, e.g. 'L:4' means layer 4.
- The grid unit for the active window, e.g. 'Grid: .1' means smallest grid unit is .1 units. (This is only visible when the grid is visible, i.e. when the grid is on and the model is not rotated).
- Four numbers that give the current location of the "view camera": Lat, Lon, Radius and Tilt.

To display menu option information on the status bar, put the pointer on a menu item to highlight it. Information associated with the highlighted item will appear on the status bar.

To toggle the status bar on and off:

Choose **View>Status Bar**.

Toolbars and Toolbar Buttons

Toolbars

Toolbar buttons are wonderful shortcuts for frequently-used menu options. Using them can save you many mouse clicks (which translates into saving wrists and fingers!)

To turn on a toolbar:

Go to **View>Toolbars** and check the appropriate toolbars.

To move a toolbar:

Click on the space at either end of, or in between, the buttons and drag the toolbar.

To float a toolbar:

Drag the toolbar away from any window edges.

Toolbar Buttons

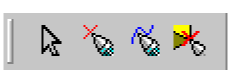
Toolbar buttons (like the menu choices for which they are shortcuts) are available only when the program "conditions are right," e.g. Edit Definition is only available when one or more entities have been selected. When a toolbar button is available, it has a colored icon; when it is unavailable, the icon is grayed out.

Some toolbar buttons execute an action immediately after being clicked (e.g. Select All). Other buttons put the mouse pointer into a "mode" (e.g. Zoom In/Out) that stays on until you click the button again or choose a different mode. These buttons have an on (pushed in = light gray background) and off state. Still other buttons are toggle switches for a display or editing aid (e.g. View>Grid or Tools>Options>Dragging>Orthogonal). These buttons work like the mode buttons: click to turn on, click again to turn off.

To execute an action with a toolbar button:

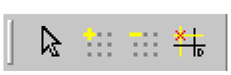
Click the appropriate button.

The Drawing Toolbar



Select Mode, Quick Point Mode, Quick Spline Mode, Digitizer Mode (if authorized for this option)

The Edit Toolbar

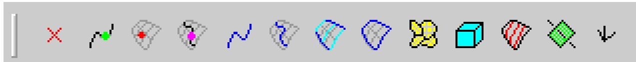


Select Mode, Set All Snaps, Clear All Snaps, Orthogonal Dragging

The File Toolbar

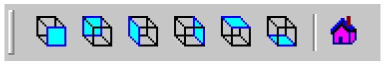
New File, Open File, Save File (New and Open not available in SolidWorks Integration mode)

The Insert Toolbar



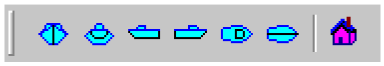
Insert Point, Bead, Magnet, Ring, Curve, Snake, Lofted Surface, Surface, Composite Surface, Solids, Contours, Planes, Frames

The Mechanical View Orientation Toolbar



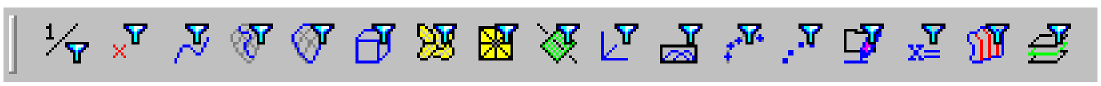
Front, Back, Left, Right, Top, Bottom, Home view orientations

The Marine View Orientation Toolbar



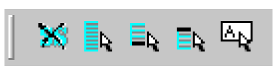
Bow, Stern, Port, Starboard, Deck, Keel, Home view orientations (Only available with Marine Options)

The Selection Filters Toolbar



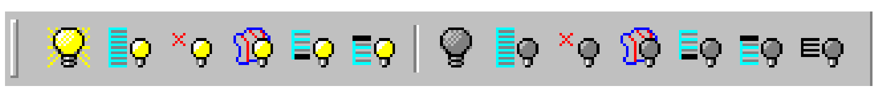
Invert Class Filters, Class Filters: Point, Curve, Snake, Surface, Solid, Composite Surface, Triangular Mesh, Plane, Frame, Relabel, Graph, Wireframe, Real, Contours, Current Layer Filter

The Selection Set Toolbar



Clear Selection Set, Select All, Select Parents, Select Children, Select by Name

The Show-Hide Toolbar



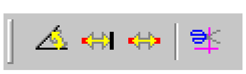
Show: Selection Set, All, Points, Contours, Parents, Children
Hide: Selection Set, All, Points, Contours, Parents, Children, Unselected

The Standard Toolbar

Windows: Cascade, Tile Horizontal, Tile Vertical

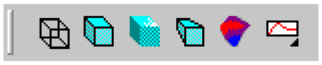
Undo, Redo

The Tools Toolbar



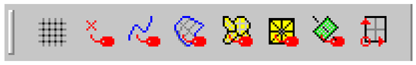
Angle, Clearance, Distance, Trim Snake

The View Display Toolbar



Wireframe, Shaded, Render, Perspective, Surface Curvature, Curvature Profile

The View General Toolbar



Grid
Nametags: Points, Curves (curves and snakes), Surfaces, Planes, Entity Orientation

The View Modify Toolbar

Previous View
Zoom to Fit, Zoom to Area, Zoom In/Out, Zoom to Selection, Rotate, Pan

Keyboard Shortcuts

SurfaceWorks offers many keyboard shortcuts (as wonderful as toolbar buttons for saving wrists!). For some, you just press a single key; for others, you hold down Shift, Ctrl, and/or Alt while you press a second key.

Some shortcuts are "toggles" — pressing the shortcut key(s) the first time turns on something that was off (or vice versa); pressing the key(s) a second time turns it off (or vice versa).

All Keyboard Shortcuts Listed Alphabetically by Option

Action	Key(s)
Edit>Delete	**Delete (or Del)** key
Edit>Redo	**Ctrl+Y**
Edit>Undo	**Ctrl+Z**
File>New	**Ctrl+N** (not available in SolidWorks Integration mode)
File>Open	**Ctrl+O** (not available in SolidWorks Integration mode)
File>Save	**Ctrl+S**
File> Component>Load	**Alt+L**
File> Component>Save	**Alt+S**
Help>Help Topics or context sensitive help	**F1**
Insert>Copy Last	**F4**
Insert>Digitizer Mode	**D**

Insert>Quick Point Mode	**Q**
Insert>Quick Spline Mode	**S**
Nudge point	**Alt + Arrow**
Select>Clear Selection Set	**Ctrl+L**
Select>Reset All Filters	**Ctrl+F**
Select>Select All	**Ctrl+A**
Select>Select Mode	**Esc**
Show-Hide>Hide Selection Set	**Ctrl+H**
Show-Hide>Show Selection Set	**Ctrl+W**
Tools>Adopt Children	**Shift+A**
Tools>Command Window	**W**
Tools>Layers	**L**
Tools>Options	**O**
Tools>Options, Dragging, Dragging, Orthogonal Drag (toggle)	**F7**
Tools>Trim Snake	**T**
View>Axes (toggle)	**A**
View>Display>Perspective (toggle)	**P**
View>Grid (toggle)	**G**
View>Marine View Orientation>Bow	**Shift+X**
View>Marine View Orientation>Deck	**Z**
View>Marine (or **Mechanical**) **View Orientation>Home**	**Home**
View>Marine View Orientation>Keel	**Shift+Z**
View>Marine View Orientation>Port	**Shift+Y**
View>Marine View Orientation> Starboard	**Y**
View>Marine View Orientation>Stern	**X**
View>Modify>Pan Pan - smaller increment	**Ctrl+arrow keys** **Shift+Ctrl+arrow keys**
View>Modify>Previous	**F8**
View>Modify>Rotate Rotate - smaller increment	**arrow keys** **Shift+arrow keys**
View>Modify>Zoom In / Out Zoom In / Out, smaller increments	**PgUp/PgDn** **Shift+PgUp/PgDn**
View>Modify>Zoom to Fit	**F**

View>Modify>Zoom to Selection	**F10**
View>Symmetry Images	**F5**
Window>Next SurfaceWorks window	**Ctrl+Shift+Tab**
Window>Previous SurfaceWorks window	**Ctrl+Tab**

Mouse Actions

In a graphics display, you can use the mouse to do many things:

In graphics displays, **clicking on an entity that can get through the filters** clears the existing Selection Set and makes the clicked entity the Selection Set. If only one entity of lowest dimensionality is at the end of the pointer, the program will select it; we call this an unambiguous entity (or unambiguous selection). If more than one entity of lowest dimensionality is targeted, you'll get a dialog box (which includes all entities available at that location) asking which entity you mean to select or edit; we call this an ambiguous entity (or ambiguous selection).

If your computer has sound, when you click on an entity that **cannot get through the filters *AND* if Tools>Options>General>Sounds>Fail Filters is turned on**, your computer will chirp (or beep) at you.

Ctrl+click toggles the selection status of the entity, either adding the entity to the Selection Set following the above rules or removing the entity from the Selection Set. If an entity is included in the Selection Set more than once (e.g. for a closed B-spline Curve), the last instance selected would be the first deselected with Ctrl+click.

Ctrl+shift+click adds the entity to the Selection Set a second (or third...) time.

Dragging a selection rectangle (pointing in whitespace, then holding the left mouse button down while dragging to a diagonally opposite corner) puts into the Selection Set all entities that are inside or crossing the selection rectangle AND that can get through the filters.

Dragging a point (pointing to a point, then holding down the mouse button while moving the pointer to a new location), moves the point to the new location specified with the mouse. Note that some dependent point entities (i.e. Projected Magnet) cannot be dragged)

Double-clicking on an unambiguous entity (that is, double-clicking when the pointer is aimed at just one entity of lowest dimensionality) displays the Edit Definition dialog for that entity. If the selection is ambiguous, indicated by the pointer, you get the Which Entity dialog. Select the entity you want, then click the Edit button to get the Edit dialog.

Rotate center mouse wheel (if available) zooms the display in or out to the location of the cursor.

Click and hold center mouse wheel or button (if available) puts SurfaceWorks into rotate mode. (This is equivalent to choosing  and clicking and holding the left mouse button).

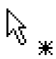
Press <Ctrl> and click and hold center mouse wheel or button (if available) puts SurfaceWorks into pan mode. (This is equivalent to choosing  and clicking and holding the left mouse button).

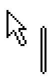
Press <Shift> and click and hold center mouse wheel or button (if available) puts SurfaceWorks into Zoom to Area mode. (This is equivalent to choosing  and clicking and holding the left mouse button).

Pointer Shapes

In the graphics window, the shape of the pointer tells you what is happening:

 Select Mode, whitespace

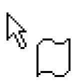
 Select Mode, aimed at a point

 Select Mode, aimed at a line

 Select Mode, aimed at a curve

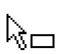
 Select Mode, aimed at an arc

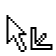
 Select Mode, aimed at a wireframe

 Select Mode, aimed at a surface

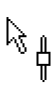
 Select Mode, aimed at a composite surface

 Select Mode, aimed at a contours entity

 Select Mode, aimed at a plane

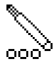
 Select Mode, aimed at a Frame

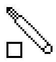
 Select Mode, ambiguous selection

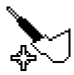
 Editing Tangent Boundary Surface, aimed at a slider control

 Quick Point Mode

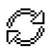
 Quick Spline Mode

 Quick Spline Mode, ambiguous selection

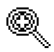
 Quick Spline Mode, over point

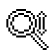
 Digitizer Mode

 Distance Mode

 Rotate Mode

 Pan Mode

 Zoom to Area Mode

 Zoom In/Out Mode

Dragging and Nudging

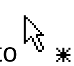
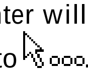
Entities You Can Drag and Nudge

Point	Bead	Magnet	Ring
	Arc-length Bead		Arc-length Ring
	XYZ Bead		XYZ Ring
Tangent Point			

When you drag or nudge a point, its coordinates are dynamically displayed on the status line. Absolute coordinates are shown for all kinds of points. In addition, parameter (t, u, v, etc.) values and relative coordinates/parameter values are displayed as appropriate. The entire model updates dynamically as you drag or nudge. When dragging, the cross-hairs show where the point actually is.

Dragging a Point

To drag a point:

1 **Aim the tip of the pointer** at the point; the pointer will change to  (If the selection is ambiguous, the pointer will change to . In this case, click to get the Which Entity dialog, select the point you want in the graphic window. The pointer will jump back to the point location. Continue the steps below.)

2 **Press and hold down** the mouse button on the point.

3 **Drag** the point to the new location, then **release** the mouse button.

Nudging a Point

To nudge a point:

1. **Aim the tip of the pointer** at the point; the pointer will change to . Click to select the point. (If the selection is ambiguous, the pointer will change to . In this case, click to get the Which Entity dialog, select the point you want in the graphic window. The pointer will jump back to the point location. Continue the steps below.)

2. **Press and hold down** the **Alt key** and then any of the **arrow** keys.

3. The point will move the distance specified in **Tools>Options>Dragging>Nudge amount**, parallel to the screen, in the direction of the arrow. Pressing shift will move the point one tenth of this distance.

Nudging values in the Properties Manager

Any real value field in the Properties Manager can now be incremented or decremented ("nudged") with the keyboard arrow keys. As shown in the images below, the value field must be active; if the <Alt> key is pressed in conjunction with either the <Up Arrow> or <Down Arrow> the value will be nudged up or down by a specified increment, depending on the value's unit dimensions.

When the real value has unit dimensions of length (for example: coordinates or offsets of Points, radius of a Polar Point or Radius Arc), the increment is the "Nudge amount" value set on the Tools>Options>Dragging tab (default 0.1 unit). This increment also applies to any Variable that has unit dimensions of length.

When the real value is a unitless parameter (for example, t parameter of a Bead, or u or v parameter of a Magnet), the increment is 0.1.

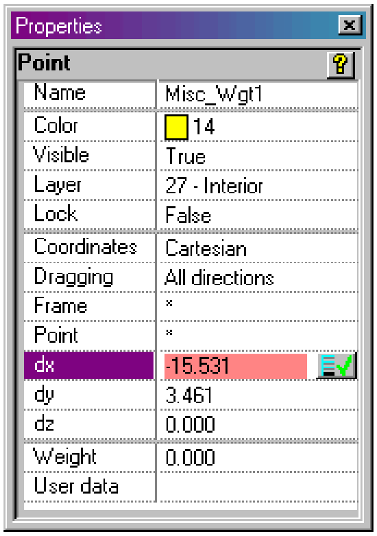
When the real value is a unitless index (for example, the First index or Last index of a Contours entity), the increment is 1.

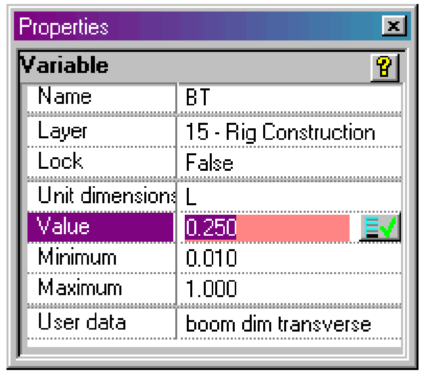
When the real value is an angle in degrees (for example, Latitude or Longitude of a Polar Point; Angle of a Revolution Surface), the increment is 10 degrees.

The Value of a Variable with unit dimensions other than length can only be nudged if the Variable has both upper and lower limits; the increment is 1/10 of the space between the two limits.

In all cases if the <Shift> key is also held down along with the <Alt> key, the nudge amount will be 1/10 of what it would be with <Alt> key alone.

The left image below shows the dx offset of a Point selected for nudging. The increment would be the "Nudge amount" set on the Options/ Dragging tab. The right image shows selection of the Value field of a Variable with units of length. The increment would likewise be the "Nudge amount" from the Tools>Options>Dragging tab.





Nudging Variable Values in Shaded or Wireframe Views

Certain variables can also be nudged in the graphic views when they are just selected, without use of the Properties Manager. To qualify for this, the following conditions must be met:

1 The selection set must consist of a single Variable.

2 The Variable must have unit dimensions of length, or must have both upper and lower limits.

3 The Variable's value must be a real constant.

4 None of the property fields in the Properties Manager can be active.

The nudge increments are the same as were detailed above for nudging in the Properties Manager.

How Points Respond to Dragging and Nudging

The various kinds of points respond somewhat differently to dragging:

- **Points**: The movement of the point is parallel to the plane of the screen. If you want to move a Point a selectable distance along one of the axial directions, use one of the standard orthographic views. If the plane of the screen is not perpendicular to one of the axes, then all three of the Point's coordinates will change.

- **Tangent Points**: The movement will be properly constrained so the entity remains on the tangent of its parent curve or snake. When dragging, the new position will be at the intersection of the cross-hairs.

- **Beads, Rings, Arc-length Beads, and Arc-length Rings**: The movement will be properly constrained so the entity remains on its parent curve or snake. When dragging, the new position will be at the intersection of the cross-hairs. You can drag or nudge a bead or a ring beyond the ends of its parent curve or snake; it will stay on the fair extension of the curve or snake, although once you get very far beyond an end, results may be somewhat unpredictable. (You can still accurately place the bead or ring by typing values into the Edit dialog or, if “Prompt after drag” is on, in the Location dialog.)

- **XYZ Beads and XYZ Rings**: While these entities are intended to be inserted right where you want them, you can drag or nudge them should the need arise. Generally, XYZ Beads and XYZ Rings respond to dragging and nudging the same way as parametric Beads and Rings, the exception being when there are multiple possible solutions. In such cases, you will only be able to drag or nudge in the general region of the current solution — you won’t be able to drag or nudge from the current intersection of the coordinate you’ve chosen to another intersection (of course, you can Edit Definition and place the entity anywhere you want).

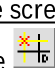
- **Magnets**: The movement will be properly constrained to the parent surface. When dragging, the new position will be at the intersection of the cross-hairs. You can drag or nudge a magnet beyond the edges of its parent surface; it will stay on the fair extension of the surface.

- **Offset Points, Mirrored Points, Projected Points, Projected Magnets, Intersection Rings**: These entities cannot be dragged or nudged, since they get their positions from the basis parent(s).

Snaps

Tools>Options>Dragging lets you specify snap increments for dragging 3D Points and XYZ Beads / Rings (X,Y,Z), Beads and Rings (t), and magnets (u,v). You can toggle the snaps on / off singly or in combination.

Orthogonal Dragging Mode

When orthogonal dragging is on, movement is constrained either to horizontal or vertical with reference to the plane of the screen. Toggle Orthogonal dragging on / off in **Tools>Options>Dragging** or with the  button.

Timesaving Tips

- Save your work frequently:

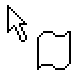
  While in SurfaceWorks, choose **File>Save** or click  to save as a .MS2 file.

  For saving in **SolidWorks Integration** mode, see the Using SurfaceWorks with SolidWorks chapter.

- Learn keyboard and toolbar shortcuts and use them.

- Use shortcut menus to reduce mouse movement.

- Cultivate the habit of reading the status bar. It presents lots of useful information.

- When selecting in a graphic window, take advantage of Select Mode's built-in "filter" for a single entity of lowest dimensionality — when you want a point, click it; when you want a curve, click it where you don't overlap a point; when you want a surface, click where you are not also on top of a curve (in wireframe display, you need to click one of the surface's interior u or v lines — the pointer changing to  lets you know when it is "seeing" the surface).

- Use the Entity Class filters and the Current Layer filter to help you select and/or edit entities. In the graphics windows, filters can help you avoid making ambiguous selections that require you to answer the Which Entity dialog. In the Available Entities pane, they can cull and shorten the list of entities displayed.

- Take advantage of the Invert Entity Classes filter. If all class filters are on, you can quickly filter for just one class of entities. E.g. to filter for points only, you would:

  1 Click the **Points filter**  **off** (yes, off — have faith; there is method to our seeming madness!).

  2 Click  to invert the entity class filter settings. Now only points can get through the filters.

- Take advantage of the Hide Unselected menu option. It can be particularly useful when you only want to display a few of the entities in your model:

  1 Select (click, then Ctrl+click) the **entities you want to display**.

  2 Click  or choose **Show/Hide>Hide Unselected**.

- Take advantage of the **Select>Invert Selection Set** menu option.

- Use the Parents and Children tabs of the Entities Manager to help you keep track of dependencies between entities. Expand children and parents to your heart's content!

- Take advantage of the Selection Set Manager. It can show you whether you have selected the entities you want and whether they are in the correct order.

- Take advantage of the Available Entities Manager. Except when you are in the support field of an Edit dialog, it only lists entities that can get through the currently set filters. This can be very helpful when preselecting parents for an entity.

- Read the User's Guide chapter "Using SurfaceWorks Effectively" and practice using the design guidelines offered therein — they are based on the collective experience of many skilled designers.

Basic Concepts

Solid modeling and Surface modeling

There are two different styles of CAD software: solid modeling and surface modeling.

The solid modeling style is characterized by 2D sketches, extrusions, revolutions, and lofts of those 2D sketches to form solids, blends on the edges of solids, and Boolean operations between solids.

The surface modeling style is characterized by control points used to define 3D curves by a variety of methods and these curves are then used to define surfaces by a variety of methods. SurfaceWorks extends this style by making each point, curve, and surface an entity that knows how it was made. This allows the designer to express design intent by building relationships between geometric entities. Any entity properties set when an entity is created can be edited at any time.

SurfaceWorks has the capability to create a wide variety of complex, freeform curves and surfaces, while at the same time maintaining relational editability and bi-directional associativity. the designer to create and maintain rich relationships between entities that express and capture design intent.

Entity Dependencies

Any engineer knows that design is rarely a linear process: starting from a set of objectives, proceeding through a series of predefined steps, and arriving at a final design. Engineering would not be a very interesting vocation if it were. In most cases design is actually a very complex iterative process: pushing through several design stages, evaluating the resulting design against the objectives, backtracking through some or all design stages, taking new paths, perhaps even going back to the beginning and redefining the objectives. Particularly in the design of 3D entities with subjective aesthetic requirements, many cycles of visual evaluation and refinement are typical. The problem is that each time you backtrack, you have to do all the design steps from that point forward all over again.

In SurfaceWorks you can establish relationships between entities that make this process easier. SurfaceWorks supports numerous kinds of relationships, such as (but not limited to):

linear distance
arc-length distance
coincident with (or constrained to) a curve
coincident with (or constrained to) a surface
mirrored location
projected location

arc tangency
G1 and G2 surface continuity

The relationships are stored in the model file, so that if one or more of the underlying entities is changed, the dependent entity (and all of its dependents, and all of their dependents, etc.) automatically change and update to preserve the relationships.

Built-in Relationships

In SurfaceWorks, relationships are built into the entity definitions. For example, by creating a bead entity, you automatically have a point that is coincident with its parent curve. This makes for a larger number of entity types to choose from, but each type of entity has a descriptive name that is easy to learn. To continue with beads, there are several types of bead entities, including the Bead entity just introduced (which is a bead located by t-parameter), the Arc-length Bead (a bead located by arc-length), and the XYZ Bead (a bead located by an X, Y, or Z value). For details about these entities, see their respective entries in the “Entity Descriptions” chapter.

SurfaceWorks also has automatic relations. Another way to create a Bead entity is to preselect a curve and click a “quick point” — a bead will be automatically created on that curve. For details, see “Creating Entities - Quick Points”.

Dependency and Exact Junctions

Dependency is often the key to achieving and maintaining exact junctions between curves and surfaces. For example, suppose ‘side’ and ‘top’ are two surfaces, each depending for one of its edges on the curve ‘seam’. If ‘seam’ is altered, both ‘side’ and ‘top’ will change in their absolute shapes, but they will continue to meet precisely and seamlessly on those edges. Or take a somewhat similar, but subtly different approach to the same junction. Suppose ‘side’ is defined first, and ‘seam’ is defined as a snake on ‘side’ — possibly lying along one edge of ‘side’. Now ‘top’ can be defined as a surface that uses ‘seam’ as one of its edges. Subsequently, you can freely modify ‘side’ and be assured that the ‘top’ will always continue to meet the ‘side’ precisely. We refer to these joins as “durable” because they are maintained through subsequent changes to the underlying geometry.

These two side-top examples achieve the same goal, but the dependencies are quite different. In the first, both ‘side’ and ‘top’ depend directly on ‘seam’:

‘seam’
    ‘side’
    ‘top’

In the second, ‘seam’ depends directly on ‘side’, and ‘top’ depends directly on ‘seam’:

‘side’
    ‘seam’
        ‘top’

Dependency can extend to many levels. For example:

’pt1’ (a point)
        ‘meridian’ (a control curve with ‘pt1’ as one of its control points
            ‘fuselage’ (a surface defined in part by ‘meridian’)
                ‘outline’ (a snake on ‘fuselage’)
                    ‘canopy’ (a surface in part defined by ‘outline’)
                        ‘window_frame’ (a snake on ‘canopy’)

'rivet' (a ring on 'window_frame')

Beads, Magnets, and Rings

Beads are points coincident with (or constrained to lie on) a curve. You might think of the curve as a thin, rigid wire; the bead is threaded onto it and can slide along. However, the bead doesn't slide around on its own — the bead is rubber and the hole is slightly undersize, so the bead has enough friction to stay where you put it. Beads are used to mark locations along curves and, in some cases, to identify a particular curve.

Magnets are points coincident with (or constrained to lie on) a surface. You might think of the surface as being formed out of sheet metal and the magnets as being the little decorative magnets you use to pin notes up on your refrigerator. You can move a magnet freely over the surface, but it stays where you put it. Magnets are used to form snakes and to mark a particular location on a surface, such as the position of a hole or a fastener.

Rings are points coincident with (or constrained to lie on) a snake. You might think of the snake as having ring markings from one end to the other — you can count the rings to specify locations along the length of the snake. Since a snake is constrained to lie on a surface, a ring on a snake is too, so a ring can serve the same functions as a magnet. Otherwise, rings are very much like beads. Rings are used primarily to mark locations along snakes and sometimes, like magnets, to define other snakes.

Parametric Curves

All of the curves you can form in SurfaceWorks also has are defined by reference to a "parameter" t, which takes the value 0 at one end of the curve (the "start") and 1 at the other end (the "end"). The rule that defines the curve from its data points is written, mathematically or in computer code, in terms of the variation of t. You can think of each point of a curve as being labeled by a specific value of t (Fig. 1). In many situations t is roughly the arc length from the start of the curve to a given point, divided by the total arc length between the curve ends.

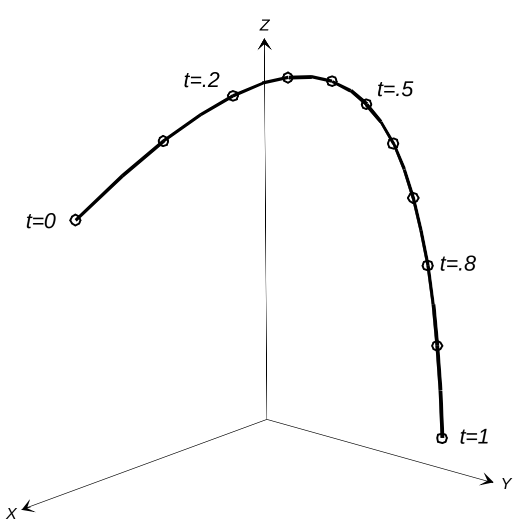


Fig. 1. A curve marked at t = .1 intervals.

It is quite accurate and appropriate to think of the generation of a parametric curve as being a continuous mapping or correspondence between the real numbers from 0 to 1 and points in 3-dimensional space (Fig. 2). You can also think of the curve as being the path of a moving point and t as being time.

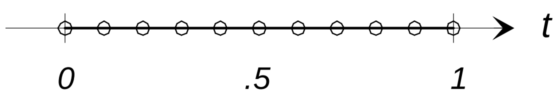


Fig. 2. A representation of the t parameter space corresponding to Fig. 1.

Actually, all the curve types used in SurfaceWorks exist beyond their nominal endpoints of t = 0 and 1. For example, you can put a bead on a curve at t = -0.1 or at t = 1.2. These beads would be points off the ends of the curve, on a logical, smooth extension of the curve itself.

Parametric Surfaces

All of the surfaces you can form with SurfaceWorks (except for Trimmed Surfaces) are defined with reference to a pair of parameters, u and v, each of which has a nominal range of 0 to 1. The rule that defines the surface in terms of its basis curves and/or points is written in terms of these two parameters. You can think of every point of a surface as being labeled with a particular pair of values for u and v. Generation of the surface is a continuous mapping from points in the unit square to points in 3D space. We refer to the unit square as the “u-v parameter space” of the surface. It is a 2D space, each of whose points corresponds to a point on the surface.

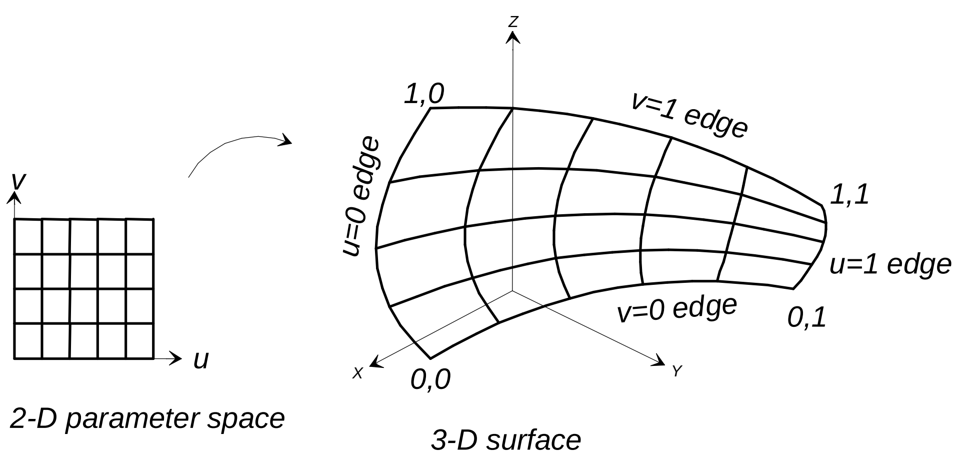


Fig. 3. A parametric surface (right) and its u-v parameter space (left).

At any point on a parametric surface there is a direction (called the “u-direction”) in which u increases while v is constant, and similarly there is a direction (called the “v-direction”) in which v increases while u is constant. The only times you usually need to be conscious of the u- and v-directions are:

- when placing magnets on the surface; these are located by their u and v parameter values
- when deciding how finely the surface should be subdivided in both directions for tabulation and display

The four edges of a parametric surface can be identified as u = 0, u = 1, v = 0, and v = 1. The four corners have (u,v) values of (0., 0.), (0., 1.), (1., 0.), and (1., 1.).

The positive normal direction of any surface (represented by a large arrow when entity orientation is displayed) is determined by using a form of the right hand rule with the u- and v-directions. With the fingers of your right hand pointing along the positive u direction and your palm facing in the positive v direction and your thumb pointing perpendicular to u and v, your thumb points in the direction of positive normal. If you select **Reverse Orientation,** in the **Property Manager,** of any surface, the positive normal direction is reversed.

Just as curves exist beyond their nominal endpoints, most SurfaceWorks surfaces extend smoothly outside their nominal parameter range of 0 to 1. A magnet placed at u = -0.1, v = 0.45 would be a point in space lying on the fair extension of the surface outside its nominal boundaries, near the middle of the u = 0 edge.

Parametric Snakes

Parametric snakes work in a way so similar to parametric curves that it is hardly worth talking about the differences. There is almost a one-to-one correspondence between curve and snake entities; that is, there are Line Snakes, B-spline Snakes, C-Spline Snakes, SubSnakes, and Projected Snakes that all work in a way very similar to the similarly named curves. The biggest difference is that any kind of snake always is constrained to lie precisely in the surface that it belongs to. Another difference is that a snake will use magnets or rings in its definition — the two kinds of point entities that always lie in the surface — where a curve can use any point entity. A third difference is that a snake generally only has its defined curve shape (Line, Arc, B-spline, etc.) in the u-v parameter space — once it is laid down on the surface, it most likely will not be a straight line, a circular arc, or a B-Spline curve in 3D space.

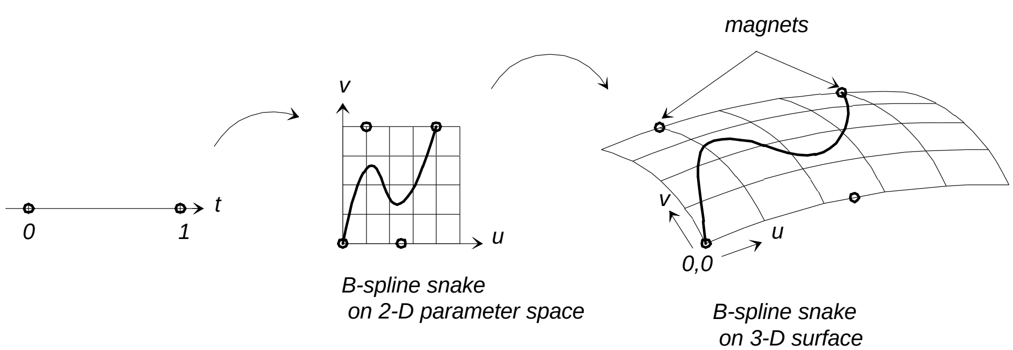


Fig. 4. A parametric snake lies in its basis surface.

Just like curves, snakes have a lengthwise parameter named t, which runs from 0. to 1.

When defining snakes we often work in the u-v parameter space of the surface — thinking about where magnets need to be located in terms of their u and v coordinates and sometimes thinking about the snake as a curve in the u-v parameter space that gets mapped point by point onto the surface.

A snake is always a valid substitute for a curve wherever a curve is used.

The Coordinate System

SurfaceWorks uses a single global Cartesian coordinate system, with coordinates X, Y, Z. A model can be placed anywhere in the coordinate system. Modeling is done directly in this 3D space (there are no 2D sketches).

Creating Entities

Overview

In SurfaceWorks, surfaces are built using curves, other surfaces and sometimes points as parents.

In this chapter, we offer some guidelines for deciding which MultiSurfSurfaceWorks entity to use for what particular task, how to find the entity creation tools, how to use the Insert dialogs, and how to use the Quick Point Mode and Quick Spline Mode shortcuts. At the end we provide “How to...” summaries for some frequently-required tasks.

Choosing Which Entity to Use

Variety and Simplicity

One of the unique aspects of SurfaceWorks is its support of a variety of different point, curve, and surface types. This variety gives you wonderful freedom to pick the right tool for each job. However, having choices does require that you *decide* — which entity do I use? Our experience suggests some general rules and guidelines. First and foremost:

The cardinal rule for choosing an entity is:

> “Choose the *simplest* entity that will do the job.”

One test of “simplicity” is (simply) “does a particular entity have specialized characteristics that do exactly what I need?” For example, if what you need is a surface to span the area between a set of three or four curves, you want one of the boundary surfaces. If in addition you want to impose tangency along one or more of those edges, the Tangent Boundary Surface is the clear choice. Surfaces made from other types of surface entities can be made tangent along one or more edges, but the Tangent Boundary Surface makes the tangency durable through most model modifications (changing supporting curves may change the tangency settings).

Another test of “simplicity” is the number of parent entities an entity requires, and whether that number is fixed or variable. A smaller number of parents means fewer entities that you have to create before you can complete the desired entity, and fewer connections to be established and maintained with the rest of the model. For example, many of the curve types will become straight lines under special conditions; but if the curve you need really is a straight line, it is simplest and best to use the Line entity: only two parents, and it’s automatically and permanently straight without further effort.

The General Process of Choosing

When you are creating a model, you will probably tend to think first about the surfaces that will make the shapes you want, then the curves needed to support the surfaces, then the points needed to support the curves — sort of opposite to the buildup of a model from points to curves to surfaces. So we'll start off by looking at surface choices, then move on to curves, points, and what to do if you change your mind.

These are general guidelines. They don't guarantee that you'll determine the perfect surface type the first time, every time; but they will give you a good place to start. Experiment! — the richness of SurfaceWorks is at your fingertips.

Choosing Surfaces by Type

Blend Surface	Curvature continuous (G2) blending between 2 surfaces.
Blister	Local protuberance on a surface.
B-spline Fitted Surface	Fit a NURBS surface to an existing SurfaceWorks surface.
B-Spline Surface	The surface emulates B-splines in the the U and V directions. Uses points as parents.
Centerpoint Boundary Surface	Surface bounded by 4 curves. A single point determines the surface bulge.
Composite Surface	Method of joining surface patches together for the purpose of flattening.
Copy Surface	Makes a scalable copy of a surface from one frame to another.
Developable Surface	Ruled surface that can be rolled out flat onto a plane without stretching.
Mirrored Surface	Mirror a surface across a plane.
NUB-spline -Fitted Surface	Fit a NURBS surface to an existing SurfaceWorks surface.
NURBS Surface	Non-Uniform Rational B-Spline Surface
Offset Surface	Offset a surface, constant offset or 4 corner blend.
Poly Surface	Joins multiple surfaces into one surface.
Procedural Surface	One point (or curve) of the procedural entity repeated to all positions of the parent beads / rings / magnets along the supporting bead / rings / magnet's host curve or surface.
Projected Surface	A surface projected onto a plane.
Relative Surface	A surface related to another. The four corner points used as parents.
Revolution Surface	Revolve a curve or shape around an axis line.
Rolling Ball Surface	Radius Fillet between Surfaces
Rotated Surface	An copy of an existing surface rotated around an axis line.

SubSurface	A subset of an existing surface. Can be smaller or larger than the original. No holes.
Sweep Surface	Sweep a curve or shape along a path. Shape stays normal to path.
Tabulated Surface	Parent point data is extacted from a table.
Tangent Boundary Surface	Surface bounded by 4 curves. Type of continuity can be assigned on each edge. Surface can be shaped using control sliders.
Translation Surface	Extrude a curve or shape along a curve.
Trimmed Surface	Trim any shape from an existing basis surface. Holes allowed.
Arc Lofted Surface	The surface emulates arc sections in the lofting direction.
B-spline Lofted Surface	The surface emulates B-splines in the lofting direction, does not pass through each curve.
C-spline Lofted Surface	The surface emulates C-splines in the lofting direction, passes through each curve.
Foil Lofted Surface	The surface emulates a Foil shape in the lofting direction. Uses 3-5 lofting curves.
X-spline Lofted Surface	The surface emulates X-splines in the lofting direction, passes through each curve.

Choosing Trimeshes by Type

Copy Trimesh	A copy of an existing Surface or Trimesh
Expanded Trimesh	Flat pattern from a 3D Surface or Trimesh
Offset Trimesh	Trimesh offset from a Trimesh or a Surface
Poly Trimesh	Joined Trimeshes; border triangles must match
Surface Trimesh	A Trimesh made from a Surface

Choosing Surfaces by Need

As a first approximation for choosing the right surface entity, try using the following question-and-answer list — just choose a surface from the list that corresponds to the first question for which you can answer “yes” (the most specialized surfaces are first because these have special characteristics to solve specific problems):

Q: Is the surface a mirror image of some other surface?

Use a **Mirrored Surface**.

Q: Is the surface an offset image of some other surface?

Use an **Offset Surface**.

Q: Can you make the surface by sliding one curve along another?

The simplest surface to use is a **Translation Surface**. In this surface the moving curve stays parallel to its initial position.

If the moving curve is to stay perpendicular to the guide curve, use a **Sweep Surface**.

Q: Can you make the surface by rotating one curve around a centerline or axis?

Use a **Revolution Surface**.

Q: Is the surface a piece of a larger surface?

A **SubSurface** is a 4-sided parametric piece you can use as a parent for other SurfaceWorks entities. A **Trimmed Surface** is a non-parametric piece that can have an irregular boundary (many sides) and can have holes in it.

Q: Do you need to produce a bulge that connects smoothly to the surface it sits on?

Use a **Blister**.

Q: Do you need to connect one surface smoothly to another?

Use a **Blend Surface** like a fillet to smooth the connection between two surfaces or a **Tangent Boundary Surface** as the second of two adjacent surfaces.

Q: Do you think of the surface as being the area inside three or four boundary curves?

Use a **Centerpoint Boundary Surface**. If you want to specify slope (tangency) and/or curvature continuity for one or more of the edges or you need to have more than one point of control over the interior of the surface, try a **Tangent Boundary Surface**.

Q: Do you want the surface to be freeform, but fixed to two edge curves?

There are several choices: use an **Arc Lofted Surface**, **B-spline Lofted Surface**, or **C-spline Lofted Surface**.

Q: Do you want the surface to pass exactly through a series of control curves?

Your surface is a **C-spline Lofted Surface**.

Q: Can you make the surface by connecting two curves with a series of straight lines?

Use a **Ruled Surface** or a **B-spline Lofted Surface** with two control curves.

Q: Can you make the surface from a series of circular arcs connecting two curves?

An **Arc Lofted Surface** will do this for you.

Q: Do you need a surface to pass exactly through a set of control points?

Try a **C-spline Lofted Surface** defined by **C-spline Curve** control curves.

Q: Do you need to make a panel that will be formed onto frames where no stretching is required, something that would use plywood in construction for example?

Use a **Developable Surface.**

Q: Do you need to make a copy of an existing surfaces in another location with a different oriention to the global coordinate system?

Use a **Copy Surface** in a **Frame.**

Q: Are none of the above surfaces seemingly correct for the application?

Use a **Procedural Surface.**

Choosing Curves

Choosing the curve to do the job generally is a fairly simple matter. If what you want is truly a line, then by all means use a Line — two points, the Line, and you're done. If what you want is actually a circle or part of a circle, use an Arc; or if what you want is a helix, use a Helix. While you can model these curves either exactly or approximately with splines, to do so you will generally need more parent points, and more fiddling with their positions. More importantly, the resulting curve will not have the durable character of a line, arc, or helix — you would need to place the control points very carefully so that the spline would take that one shape (line, arc, helix) out of all the others it's capable of being molded into.

Q: Is the curve you need identical to another existing curve, but with reversed t-orientation?

Use a **SubCurve** between two beads at t = 1 and t = 0.

Q: Is the curve you need identical to another existing curve, but with a different distribution of the parameter t ?

Use a **SubCurve** with three or more control beads, the first and last ones at the ends of the basis curve.

Q: Is the curve you need identical or closely related in shape and position to another existing curve?

Try one of these: a **Relative Curve** is formed by offsetting points from a basis curve; a **Projected Curve** is the projection of a curve, usually onto a plane; a **Mirrored Curve** is the reflection of a given curve; a **Projected Snake** can be used to establish a draft angle for mold releases.

Q: Is the curve you need a portion of an existing curve?

This is a **SubCurve** between two beads.

Q: Do you need to link up two or more curves end-to-end to make a single curve out of them?

Then a **PolyCurve** is required.

On the other hand, if your needs are not as specialized as the above curves, chances are you need to be designing with some form of spline curve:

A **C-spline Curve** passes through all its control points and is completely shaped by their locations.

A **B-spline Curve** passes through its two end control points but generally does not pass through the other ones; instead, it acts as if attracted to the intermediate control points, or attached to them with springs. B-spline Curves have lots of nice properties that make them very popular for freeform curve design. Especially when we want to control the end directions of a curve, or shape a curve that has an exactly straight portion, or has a large variation in curvature along its length, we think first of a B-spline Curve.

Choosing Points

Choosing the right point entity is usually a matter of identifying any special properties required. First, points are in four classes that reflect restrictions on their positions:

beads are constrained to lie on a curve
magnets are constrained to lie on a surface
rings are constrained to lie on a snake
Trimesh Magnets are costrained to lie on a Trimesh_D2HLink_86971
Trimesh Rings are constrained to lie on a Trimesh Snake_D2HLink_86973
the rest are "general 3D **points"**

Beads - because their only constraint is that they must lie on a curve and snakes are considered curves - can be placed on snakes. However, we don't recommend this. Because it must lie on a snake which must lie on a surface, a ring gives more information about its location. Always using rings on snakes will assure that you get the most out of the relationships that you have taken the trouble to build.

One main idea we'd like to reinforce here is the great usefulness of relative points of all kinds (3D, beads, rings, magnets). By definition a relative point always stays in a specified position relative to its parent point. By using one point with default Parents and several to many points with relative coordinates to define another entity, or even a set of entities, you can then move the entire entity or the entire set of entities by moving a single point.

Projection points (**Projected Point**, **Projected Magnet**) result from constructions that require solving for an intersection. They are more complex and in some cases more "fragile" than other points — for example, it may be that no intersection or that multiple intersections occur — but for their specialized purposes they are indispensable!

Changing Your Mind

With some planning and forethought, you'll generally get off on the right track and proceed to a satisfactory model. However, you can't always be sure how a particular surface is going to turn out until after you've made it. Or maybe you become aware of an additional requirement that you weren't thinking of at the outset.

To change points from default Parents to non-default Parents (or vice versa):

Simply Edit Definition for the point. If changing to non-default Parents, specify the Parent(s) if changing to default Parents, click in the Parent box you want to change and select "Default object" from the Surface Manager pane. SurfaceWorks asks if you want to "Preserve Absolute Location" or "Preserve Offset Values".

To change to a different curve or surface entity:

For curves or surfaces, the presence or absence of dependents makes a difference:

- If the entity has no children, you can just hide it, build the new one and see how that looks; if you like it, delete the old one and keep the new.

- If the entity has children, for example points and curves drawn on it (beads, magnets, rings, and snakes), or other curves or surfaces joined to it, you can make the new entity (to maintain model-wide relationships you may want to use some or all of the old entity's parents), then use **Tools>Adopt Children** to swap the old entity's children to the new one.

The Entity Creation Tools

To create an entity, you specify its definition in the Property Manager.

To Insert dialog an entity, you can:

- Go to the **Insert menu** and select the kind of entity you want

- Use the **Insert toolbar buttons** to access the Insert submenus for the kind of entity you want

The creation of entities in SurfaceWorks is initiated by using either the Insert menu or the appropriate Insert toolbar button. When you click an Insert toolbar button, you get a drop down menu (a button for each entity would mean far too many buttons). There are so many entity choices in SurfaceWorks because relationships are built into SurfaceWorks entities, therefore when you choose an entity you are choosing the relationship as well.

The Insert Toolbar

Each Insert toolbar button accesses a group of entities:

Toolbar button	Options on drop-down menu
Insert Point	Point, Blend Point, Copy Point, Intersection Point, Mirrored Point, Offset Point, Projected Point, Rotated Point, Tabulated Point, Tangent Point
Insert Bead	Bead, Arc-length Bead, Copy Bead, Intersection Bead, Proximity Bead, XYZ Bead
Insert Magnet	Magnet, Copy Magnet, Intersection Magnet, Projected Magnet, Proximity Magnet, Tangent Magnet, XYZ Magnet
Insert Ring	Ring, Arc-length Ring, Copy Ring, Intersection Ring, Proximity Ring, XYZ Ring
Insert Curve	Arc, B-spline Fitted, B-spline, C-spline, Conic, Copy, Expanded, Foil, Helix, Line, Mirrored, NUB-Fitted, NURBS, Offset, Poly, Procedural, Projected, Radius Arc, Relative, Sub, Tabulated, X-Spline.
Insert Snake	Arc, B-spline Fitted, B-spline, C-spline, Copy, Edge, Foil, Geodesic, Intersection, Line, NUB-Fitted,

	NURBS, Poly, Procedural, Projected, Relative, Sub, UV
Insert Lofted Surface (button with cyan control curves)	Arc Lofted, B-spline Lofted, C-spline Lofted, Foil Lofted, X-Spline
Insert Surface (button with all blue boundary)	Blend, Blister, B-spline, Centerpoint Boundary, Copy, Developable, Fitted, Mirrored, NUB-Fitted, NURBS, Offset, Procedural, Projected, Revolution, Rotated, Ruled, Sub, Sweep, Tangent Boundary, Translation, Trimmed
Insert Composite Surface	Composite Surface

When you choose a specific kind of entity from a drop-down menu, it is displayed in the Property Manager in Insert mode.

The Drawing Toolbar

There also are shortcuts for creating several kinds of point entity and B-spline and C-spline curves. These shortcuts are on the Drawing toolbar:

Quick Point Mode	To create Points, Beads, Magnets, Rings
Quick Spline Mode	To create B-spline and C-spline curves and snakes
Digitizer Mode	Creates points using the Digitizer

For details about creating entities in these modes, see "Quick Points and Quick Splines" on page 71.

The Insert Menu

The Insert menu options include the major classes of entities — Point, Curve, Snake, Surface, Frame, Plane, Wireframe, Contours, Entity List and Relabel — plus Copy Last, which uses the most-recently-created entity as a template for creating a new entity of that kind. The class options open into submenus of all the entities in the class. When you choose a specific kind of entity from a drop-down menu, it is displayed in the Property Manager in Insert mode.

The other three options on the Insert menu are the Quick Point Mode and Quick Spline Mode toggles and Digitizer Mode. For details about creating entities in these modes, see "Quick Points and Quick Splines" on page 71 and the chapter "Using the Digitizer".

Inserting Entities with the Properties Manager

In the Properties Manager, the properties and (except for pre-selected entities) the parents listed are defaults, some of which you may want to keep and some of which you may want (or need) to change as you define your new entity.

Defaults

Program-generated Defaults for Properties

In the Property Manager, defaults for properties are program-generated:

Names. Default names are made by three rules, applied as follows:

1 If the name of the last entity made for this entity ends with a number, SurfaceWorks tries to make a new name by incrementing the number.

2 Otherwise SurfaceWorks tries to make a new name by appending the digit “1”.

3 If the first two rules fail, or if this is the first instance of this entity, SurfaceWorks makes a name characteristic of the entity class (e.g. pt for Point entities, surface for Surface entities), followed by the first available number.

SurfaceWorks guarantees that default names offered do not duplicate existing names.

Color, Visibility (but not hidden status), Divisions, Type, Degree, etc. Defaults for these properties are based on the entity class or on a previously created entity of the same class.

Layer. Default layer is the Current Layer (set in **Tools>Layers, Current Layer**), unless there is a single entity in the Selection Set AND that entity is the same kind as the one being created — in which case, the new entity will be put on the same layer as the preselected entity.

Weight. Default for weight is 0.

Orientation. Default for Reverse Orientation is Normal Orientation.

Relabel. Default for relabel is * which means the default relabel for that entity is used.

Offsets, Location. The default location for a Point, Bead, Magnet, or Ring depends on the contents of the Selection Set when the point entity is created:

<u>Point</u>:

All Points have two parents, Point and Frame. In the Insert Dialog box, the default offsets are always 0,0,0, but the Point and Frame to which the offsets are relative to change depending on the entity(s) that are pre-selected.

- When the Selection Set contains no Point or Frame entity, the offsets are relative to the origin (Point) of the global coordinate system (Frame). (This is indicated in the Parents section of the dialog by a star.)

- When the Selection Set consists of only a Frame, the pre-selected entity is used as the Frame Parent. The Point Parent remains the default point, 0,0,0, of that frame.

- When the Selection Set consists of a single Point entity, the pre-selected entity is used as the Point Parent. The Frame remains the global coordinate system.

- When the Selection Set consists of a Point and a Frame entity, the pre-selected Point entity is used as the Point Parent and the pre-selected Frame entity is used as the Frame Parent.

- When the Selection Set contains two or more Point entities, the first entity is used as the Point Parent and the second defines the origin of a frame that is parallel to the global coordinate system.

Bead, Ring, and Magnet:

- When the Selection Set consists of a single Bead / Ring / Magnet entity (that is, a Bead when you are creating a Bead, etc.), the preselected entity is used for "Relative to" and the default location of the Bead / Ring / Magnet being created is the location of the preselected Bead / Ring / Magnet.

  Note: A ring can serve as a magnet. Therefore, if a ring is selected and you insert a Magnet, it uses the ring for "Relative to" and location defaults to du=0, dv=0.

- When the Selection Set does not contain a single Bead / Ring / Magnet entity (that is, a Bead when you are creating a Bead, etc.), the default location of the Bead or Ring being created is at t = 0.5 of the parent curve or snake, and the default location of the Magnet being created is at u = 0.5, v = 0.5.

  Note: You can put both rings and beads on snakes, but you can put only beads (no rings allowed) on curves.

Locked/unlocked status: Use the Locked dropdown to specify the locked / unlocked status of the entity. The default status for a newly-created entity is unlocked (False).

What does locking an entity mean?

Locked entities are entities which can be displayed (if their layer is on) and selected but only their locked status can be changed — none of the entity's other attributes can be edited while the entity is in the locked state.

Locking does NOT imply locking the real space position of entities whose shape or location is based on a relationship to other entities — i.e. moving a parent would result in movement of a locked, relational descendent of that parent. Examples:

- If you lock a Point that has a non-default Point Parent, then move its Point parent, the Child Point will move accordingly (it will maintain its specified relationship to the Parent Point).

- If you lock a B-spline Curve, then move one of its Control points, the B-spline Curve shape will update accordingly (it will maintain its specified relationship to its Control points).

Defaults for Parents

In the Property Manager, the defaults for parents may be:

- The placeholder *EMPTY* (which you must replace with one or more proper parents)

- An entity name
- A list of entity names (note that a list may be incomplete; that is, there may not be enough entities specified to meet the parent requirements for the entity)

An entity can become a default parent if it is in the Selection Set when you choose an Insert menu option. (When you select parents before you choose an Insert option, we call this “preselecting” parents.)

SurfaceWorks first looks to see if there are any entities in the Selection Set. If there are, SurfaceWorks looks at the first entity in the Insert dialog for the entity being created and tries to match it starting with the first entity in the Selection Set. If there is no match, SurfaceWorks tries the next entity in the Selection Set and so on until it finds a match, or runs out of entities to try. If there is no matching entity from the Selection Set, it places an *EMPTY* in the box for that parent and moves on to the next required parent. If there is a match found, it places that entity in the box for the parent, and moves onto the next parent. When a member of the Selection Set is chosen as a parent, it cannot be chosen again for that entity being created.

If there are no entities in the Selection Set, SurfaceWorks then looks to see if that entity has been created yet in this session. If it has, SurfaceWorks uses the parents from the last time it was used as the new defaults.

If that entity has not been used yet, SurfaceWorks uses *EMPTY* as the parents.

Note: Re preselection and Projected Points, Mirrored Points, and Projected Magnets: Because you cannot use one of these entities as a basis for another entity of the same type, if you preselect a Projected Point when creating a new Projected Point, a Mirrored Point when creating a new Mirrored Point, or a Projected Magnet when creating a new Projected Magnet, SurfaceWorks automatically uses as default the parent of the original point or magnet, thereby overriding the default use of the preselected entity as the parent.

Always check default parents to be sure they are what you want —SurfaceWorks can’t distinguish between an entity that was intentionally preselected and one that was “just left” in the Selection Set and happens to qualify as a parent for the entity being created.

Internally-defined Entities

SurfaceWorks contains several internally-defined (or predefined) entities you can use as parent entities without having to create them yourself. When you are creating an entity, you may choose these as parents (where appropriate). Notice that '*' can be used for a number of default entities. The context of its use makes it clear which entity it represents.

Note: Internally-defined entities are never visible in the drawing; you must select them from the Surface Manager or the Available Entities Manager.

Name	What It Is
* (Default object)	point located at 0,0,0 of a frame
* (Default object)	frame (global coordinate system)
* (Default object)	relabel (default relabel for entity)

*0	bead or ring at t=0 end
*1	bead or ring at t=1 end
*X=0	X=0 plane
*Y=0	Y=0 plane
*Z=0	Z=0 plane
*UNIFORM	knotlist
*00	graph, relabel, or system magnet
*01	graph, relabel, or system magnet
*10	graph, relabel, or system magnet
*11	graph, relabel, or system magnet

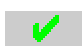
Note: Internally-defined entities do not shift or rotate. If you plan to modify your model in either of these ways, make your own entity at the appropriate location.

Using the Properties Manager

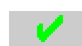
Overview

To create an entity once you have it in the Properties Manager:

1 **Edit any of the default data** you want, e.g. name, color, parents, etc. Note that you must replace any *EMPTY* placeholders.

2 Click  to **OK**.

The new entity becomes the currently selected entity.

If you **Cancel**  an Insert without having clicked , no entity is created.

To reverse the creation of an entity:

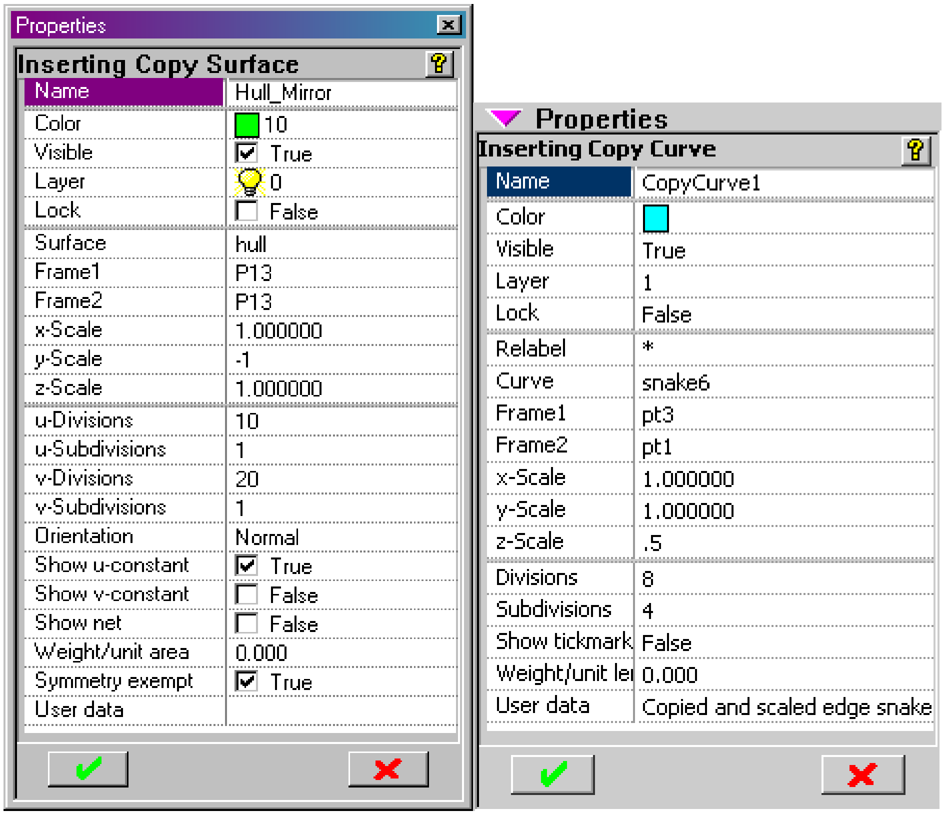
If it's still in the Selection Set you can either press <Delete> or chose **Edit>Delete** (it will have no children, if it was the last entity you created); or you can click  or use **Edit>Undo**.

Specifying Properties and Parents

Data defining the entity being inserted are presented in the Properties Manager. For explanation of a particular entity's characteristic properties and parents (the data, other than name, on the General tab), see its entry in "Entity Descriptions". For explanation of properties common to certain classes or types of entity, see "Understanding SurfaceWorks Entities - Defining Entities".

Contents of the individual tabs in the dialog are described below.

Properties Manager



For explanation of a particular entity's characteristic properties and parents (the data, other than name), see its entry in Entity Descriptions.

Name. To edit the default offered for entity name, click in the name field and make your changes. Refer to "Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Name " for details on valid entity names.

Color. Select the entity color from the drop down color palette in the dialog.

Visibility. Drop down menu to set visibility, True or False, see "Understanding Entities - Defining Entities - Properties Common to All Entities - Visibility".

Layer. Set the layer by choosing it from the drop down list. Use **Tools>Layers** to name layers, and to turn the display of individual layers on and off.

Locked. A locked entity can not be edited or moved.

Characteristic Properties. Characteristic properties are entity-specific properties such as degree for B-spline Curves, type for Arc, pitch for Helix, draft angle for Projected Snake, style for Blister.

Relabel. Available for all curves, snakes and lofted surfaces, and some other surfaces. Allows you to relabel how the parameter t is distributed along a curve or snake or u and v are distributed within a surface.

Parents. Parents of an entity, if required, may consist of:

- One or more individual parent entities (e.g. an Offset Point requires just one individual parent: Magnet, while a Projected Point requires two individual parents: Point and Mirror). In the dialog, there is an individual field for each of these.

- One or more lists of a variable number of parent entities (e.g., a list of control points for a B-spline Curve or a list of control curves for a lofted surface). In the dialog, these appear in a list box.

To specify parent entities:

Click anywhere in the parent data box. The box's background changes to a pink and you are in parent specification mode:

- The Selection Set pane now lists any preselected parents or *EMPTY*.
- The Available Entities pane is constrained to list only the entities that qualify as parents.
- Any filters you had previously set are temporarily overridden.

You can select parents by clicking (and Ctrl+clicking and Ctrl+Shift+clicking if required) in the graphics display, the Available Entities pane, or the Surface Manager. For details on how to select entities, see "Fundamentals - Selecting Entities".

Note: In the Entity Manager, the icons for currently-selected entities are shown with a cyan rather than a yellow background. You can right-click in whitespace in the Entity Manager and choose Expand to Selected Entities to see them. In the Available Entities Manager, any entities selected are highlighted.

Divisions. Set divisions and subdivisions for curve and surface entities. t divisions and subdivisions are used for curves and snakes. u and v divisions and subdivisions are used for surfaces. The higher the divisions and subdivisions, the more accurately the entity will be evaluated (at the cost of processing speed). Accuracy also can be improved by coordinating divisions between parent and child entities. For further details on divisions, see "Understanding Entities - Defining Entities - Additional Curve and Snake Properties and Additional Surface Properties".

Reverse Orientation. Available for all surfaces. Reverses u and v directions when applied.

Visibility Options. Set visibility options using drop down menus. Options available depend on the type of entity being created. For explanation of the various visibility options, see "Understanding Entities - Defining Entities - Properties Common to All Entities - Visibility".

Weight. Set the weight (point), weight per unit length (curve) or weight per unit area (surface) of the entity.

User Data. User Data is a new attribute, editable in both the Property Manager and Multiple Edit. A text string of up to 40 characters can be entered here.

Quick Points and Quick Splines

Quick Point Mode: In this mode, you simply click locations for one or more Point, Bead, Magnet, or Ring entities. When you end the series, the entire set of points is accumulated, in order, in the Selection Set — ready for you to go on to making a curve (or snake), should you so desire.

Quick Spline Mode: In this mode, you create B-splines or C-splines (curves or snakes) by dragging out spline segments or clicking control points; spline preview draws as you move the pointer on the screen.

Quick Points

Shortcuts:  or **Q**

The quick points function is a shortcut for making points (Point, Bead, Magnet, and Ring entities), and it leaves you ready to insert a curve (or snake) parented by the set of points. In this drawing mode, called Quick Point Mode, you create point entities by simply clicking locations in the graphics display.

Which kind of point entity will be created is determined by the contents of the Selection Set when Quick Point Mode is invoked. To make Points in space, preselect one or more point entities or nothing; to make Beads, preselect a curve in space; to make Magnets, preselect a surface; to make Rings, preselect a snake.

Since Quick Point Mode operates in 3D space (rather than on a 2D sketch), the 3D location of quick points is determined by a combination of the pointer’s 2D location on the screen plus depth information from a preselected entity or, for Points in space, a default plane. The points automatically are given a series of default names which follow the general rules for naming entities.

Setting the Depth for Quick Points

Quick points in space are Point entities with default Parents. The depth for 3D points is taken from a default plane parallel to the screen and passing through either the preselected point, the last point in the Selection Set (if there is more than a single point preselected) or through the origin of the coordinate system (if no point is preselected). If one or more point entities are preselected, they will become the first points in the cumulative Selection Set.

Quick beads, magnets, and rings are (absolute) Bead, Magnet, and Ring entities. The location of the quick points will be the closest point on the host entity to the mouse click on the screen. All the Beads, Magnets, or Rings you place in any quick point series will have the same host, e.g. all the quick magnets in a series will lie on the same surface (i.e. the surface that was preselected).

To create quick points:

1 **Preselect** an appropriate entity for the kind of points you wish to create:

- If you want to make Points in space, preselect nothing (to use the default plane for depth) or one or more point entities (of any kind; the last point will set the depth).
- If you want to make Beads, preselect a curve in space.
- If you want to make Magnets, preselect a surface.

- If you want to make Rings, preselect a snake.

2 Click , or choose **Insert>Quick Point Mode**, or press **Q**.

3 **Click** as many point locations as you want. Each new point will be added to the cumulative Selection Set.

4 To exit Quick Point Mode, **right-click and choose another mode** (e.g. Select Mode), or press **Esc**, or click again, or choose **Insert>Quick Point Mode** again or press **Q** again.

5 Since all the points in a quick point series are selected when you end the series, **you are all set up to use the toolbar buttons or the Insert Menu to create a curve (or snake) from them**, should you so desire.

Quick Splines

Shortcuts: or **S**

The quick spline function is a shortcut for making B-splines and C-splines (curves or snakes) in the graphics display. In this drawing mode, called Quick Spline Mode, you make a spline and its parent points at the same time. As you drag (standard creation method) or click (alternate creation method) the parent points, a dynamic preview of the B-spline or C-spline that is defined by them is displayed. Defaults for the B- vs. C-spline option, for the Degree of the spline, and for the spline creation method are set on the **Tools>Options>Entity** tab.

Whether curves in space or snakes constrained to a surface will be created is determined by the contents of the Selection Set when you turn on Quick Spline Mode. To make snakes, preselect a surface; to make curves in space, preselect any kind of non-surface entity or nothing. The parent points created for curves in space are Point entities and those created for snakes are Magnets.

Since Quick Spline Mode operates in 3D space (rather than on a 2D sketch), the 3D location of the parent points is determined by a combination of the pointer's 2D location in the plane of the screen plus depth information from a default plane or a preselected point or surface entity. The points and spline automatically are given default names.

To set options for Quick Spline Mode:

Use the **Tools>Options>Entity tab** to set defaults for Quick Spline Mode:

1 Choose whether to create **B-spline Curves or C-spline Curves** (option buttons)

2 Specify the **Degree** property for the curve (1 = linear, 2 = quadratic, 3 = cubic, etc.)

3 Check the box if you would like to use the **alternate spline creation method**

Spline Creation Method Options

There are two options for how the cursor will function while in this Quick Spline Mode. In either option, when the cursor is over a point it changes to  and clicking here will make the existing point part of the Selection Set, not another quick point.

The "standard" method - click-and-drag: Press the mouse button down where you want the first parent point, drag to the location of the next point, release the mouse button. Press the mouse button down again at the same location where you released the mouse button, then drag to place the next point, release the mouse button, and so on. To finish the spline, click anywhere in the drawing other than the location of the last point or choose another cursor mode.

Alternate spline creation - series of clicks: Click the mouse button at the location of each control point, except double-click to drop the last control point (this ends the spline).

Setting the Depth for Quick Spline Parent Points

Quick splines in space: The parent points created for quick splines in space are Point entities with default Parents. The depth for the parent points is taken from a default plane parallel to the screen and passing through either the preselected point, the last point in the Selection Set (if there is more than a single point preselected) or through the origin of the coordinate system (if no point is preselected).

Quick snakes: The parent points created for quick snakes are (absolute) Magnet entities on the preselected surface. The location of the quick magnets will be the closest point on the host surface to the mouse release (or click) on the screen. All the parent magnets created for a quick snake will lie on the same host surface, and any pre-existing magnets you want to include as parents for that quick snake must also lie on that same host surface.

To create quick splines:

1 Use the **Tools>Options>Entity tab** to set the defaults for the **kind of spline entity** to be created, **Degree** for the spline, and the **spline creation method**.

2 To make a spline curve in space, you can **preselect a point entity** to set the depth for the parent Points.

   To make a spline snake, **preselect the surface** the snake is to be constrained to lie on.

3 Click , or choose **Insert>Quick Spline Mode**, or press **S**.

4 If you are using the "standard" creation method, p**oint the cursor where you want to begin the spline and press the mouse button**.

   If you are using the alternate spline creation method, see the Note below.

5 **Drag out the first segment** of the spline and **release** the mouse button. If you release at an existing point (or qualifying magnet, for snakes), that point will be used as a parent and no new point will be created at that location.

6 **Press over the endpoint, drag out the next segment**, and **release** the mouse button.

7 **Repeat dragging out segments until the spline is complete**, then to end the spline, **click any location other than the last parent point** (or exit Quick Spline Mode).

Note: If you are using the alternate spline creation method (Tools>Options>Entity tab), simply click each parent point location, except double-click the last point location (or exit Quick Spline Mode at that location) to end the spline. If you click at an existing point (or qualifying magnet, for snakes), that point will be used as a parent and no new point will be created at that location.

---

8 To exit Quick Spline Mode, **right-click and choose another mode** (e.g. Select Mode), or press **Esc**, or click  again, or choose **Insert>Quick Spline Mode** again or press **S** again.

How to…

Introduction

SurfaceWorks entities give you the ability to create a rich variety of surface shapes, complete with durable relationships. To help you discover ways in which you can capitalize on them, we've collected some brief "how to..." examples that introduce important SurfaceWorks-specific modeling techniques. (For a detailed, hands-on tour of SurfaceWorks capabilities, we recommend you run the SurfaceWorks tutorials.)

Lofting Surfaces: a Different Approach

When lofting surfaces, particularly for conceptual design, consider lofting from longitudinal curves as opposed to transverse sections and guide curves.

During the conceptual design phase of a product, generally, the focus is on the profile lines rather than the cross section data. In most cases, you can use the SurfaceWorks B-spline Lofted Surface to produce the shape using the profile lines alone. Here is an example:

We generated the toothbrush body from just the top, side, and parting line curves.

Using Curves and Snakes to Join Surfaces Exactly and Durably

One of the main features of SurfaceWorks surfaces is the ability to join them durably with a common parent along an edge. In many cases, you can accomplish this simply by using a single 3D curve as an edge of each of two adjoining surfaces. But what if the surface edge you want to join doesn't have a parent curve along it? Or what if the adjoining surface doesn't attach along the edge at all (i.e. it attaches to the interior of the surface)? In situations like this, you can insert a SurfaceWorks "snake," a curve constrained to lie on a surface, to make the join.

- An Edge Snake runs the full length of a surface's edge.
- A UV Snake runs the whole length (or breadth) of a surface at a constant u- or v-parameter value.
- A B-spline Snake or C-spline Snake can be used to make free-form curves constrained to lie on a surface.
- If only part of a snake (e.g. part of an Edge Snake or UV Snake) is needed as a parent for another surface, you can make a SubSnake of the desired length on the too-long snake.

Basic steps:

1 Create the first surface.

2 Make a snake that will serve as an edge for the surface you want to join to the first surface.

3 Create the second surface, using the snake as one of its parents.

Decreasing Surface Export Time

You can greatly decrease the IGES file generation time (and make smoother surfaces) if you "coordinate" surface divisions with the divisions of the parent curves.

For example, when creating a lofted surface, use the same number of divisions on each of the master curves, and also make sure that the master curve t-divisions x t-subdivisions is the same as, or a multiple of, the surface u-divisions x u-subdivisions. E.g. you might use:

t-divisions = 8x4 and u-divisions = 8x4 (curve and surface divisions are the same)
or
t-divisions = 8x4 and u-divisions = 4x1 (curve divisions are a multiple of surface divisions)

Similarly, when creating a Centerpoint Boundary Surface, a Tangent Boundary Surface, or a SubSurface, use the same number of divisions on curve/snake1 and curve/snake3 and make this number the same as, or a multiple of, the surface u-divisions x u-subdivisions. Do likewise for curve/snake2, curve/snake4, and surface v-divisions x v-subdivisions. For a Revolution Surface, coordinate the divisions of the meridian curve with those of the surface u-divisions x u-subdivisions; etc.

Note: Coordinating divisions does not mean that the number of u-divisions has to be the same as the number of v-divisions.

Degrees of Continuity Between Surfaces and Between Curves

A major consideration in assembling different surface entities to build a complete solid model is the degree of continuity required between the various surfaces. This applies to curves, too, and the concept is a little easier to grasp if you first think about continuity of curves.

Continuity between curves: G0, G1, and G2

G0: Two curves that join with an arbitrary angle at the junction are said to have G0 continuity, or "geometric continuity of zero order."

G1: If the curves join with zero angle at the junction (the curves have the same tangent direction) they are said to have G1 continuity, first order geometric continuity, slope continuity, or tangent continuity. Examples:

- A Line and an Arc that end normal to the plane through their shared end point (Fig. 1).

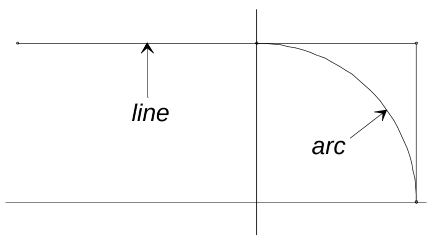


Fig. 1

- Two B-spline Curves that share an end point and where the next-to-end point of the second of the B-splines is the Mirrored Point of the next-to-end point of the first mirrored across the shared end point (Fig. 2). See also Tangency and Mirrored Points in the Building Tangency with B-splines tutorial.

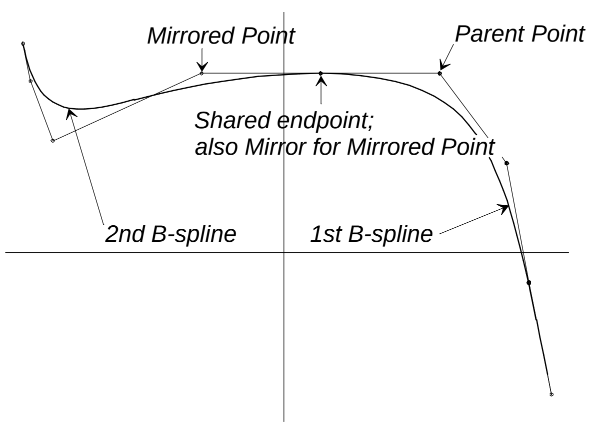


Fig. 2

- A B-spline loop that begins and ends at a Bead on a Line between the two next-to-end control points (Fig. 3). See also Tangency and B-spline Curve Loops in the Building Tangency with B-splines tutorial..

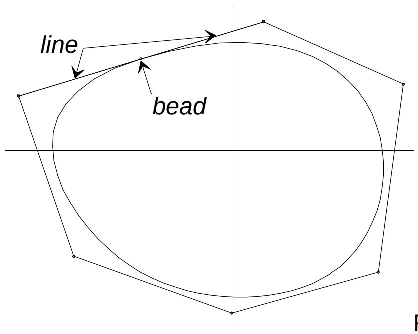


Fig. 3

G2: If the curves join with zero angle and have the same curvature at the junction, they are said to have G2 continuity, second order geometric continuity, or curvature continuity. This is relatively difficult to achieve with splines, but here are a couple of examples:

- A B-spline Curve that ends normal to a plane and its Mirrored Curve across that plane (Fig. 4). See also Tangency and Mirrored Curves in the Building Tangency with B-splines tutorial.

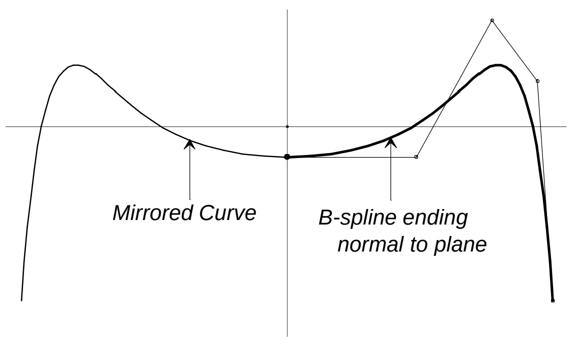


Fig. 4

- A Line and a degree-3 B-spline Curve whose ending 3 control points lie on the extension of the Line (Fig. 5).

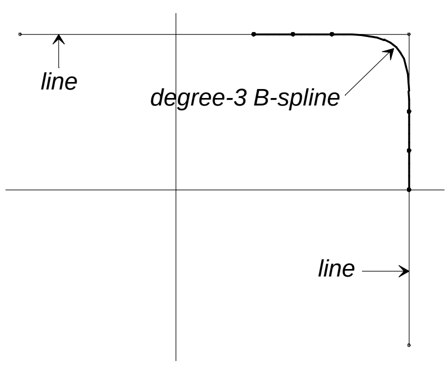


Fig. 5

Continuity between surfaces: G0, G1, and G2

G0: Surfaces that join with an angle (that is, they have different normal directions) at the junction have G0 continuity.

G1: Surfaces that join with the same normal direction at the junction have G1 continuity.

G2: Surfaces that join with the same normal direction and the same curvatures across the junction have G2 continuity.

The higher the degree of continuity, the smoother the junction will appear. G0 continuity is relatively easy to achieve and is often used in "industrial" contexts when a sharp corner does not interfere with function. G1 continuity is more trouble to achieve, and is widely used in industrial design when rounded corners and fillets are functionally required. G2 continuity, still more difficult to attain, is required for the highest level of aesthetic design, such as in automotive and yacht exteriors.

Imposing Tangency Between Surfaces

When you need to span an area with a surface that gives you control over tangency conditions along adjoining or more than two edges, try a Tangent Boundary Surface. This entity allows you to specify that the surface be:

- normal to the plane of a planar edge curve
  or
- tangent to an adjacent surface along a snake

The Tangent Boundary Surface requires 4 bounding curves / snakes (one or more of which can be a point entity, to form a 3-sided or 2-sided patch) for instance, if you want to enforce tangency along an edge, the edge must be defined either by a planar curve (but not a line) or by a snake.

Edge continuity can be either G1 (continuous slope) or G2 (continuous slope and curvature).

Note: You can also use the Tangent Boundary Surface with no edge continuity specified.

When you need to span an area with a surface that gives you control over tangency conditions along opposite edges, try a Blend Surface

Making Domes

SurfaceWorks has several alternative entities which provide a wide range of possibilities for domed shapes:

The **Centerpoint Boundary Surface** extends upward from the center of an enclosed curve. The apex of the Centerpoint Boundary Surface dome can be off-center.

Another option for domes is to create a set of profile curves and span them with **Tangent Boundary Surfaces**. In this method a dome would be separated into 4 quadrants, one or more spanned with Tangent Boundary Surfaces, any remaining ones with Mirrored Surfaces.

If the dome in question is defined by arcs in one direction it could be constructed with an **Arc Lofted Surface**.

The SurfaceWorks **B-spline Lofted Surface** is yet another method to construct a dome. Here’s an example:

Basic steps using a B-spline Lofted Surface:

1 Create an outline of the base of the dome.

2 Create a relative curve at the exact height of your dome. Be sure there is a point in the same plane as this relative curve. This point will be the apex of the dome.

3 Select the outline curves in order, bottom to top, with the last selection being the point at the apex.

4 Insert a B-spline Lofted Surface. This dome can be edited by changing the shape of the outline, the offset distance between the two curves or the location of the point within the plane of the top curve.

Editing Models

Introduction

This chapter describes how to edit individual entities and how to fair curves and surfaces.

Editing in SurfaceWorks

Editing a Single Selected Entity

There are a number of ways to Edit the properties an entity:

- In the Surface Manager, or the Available Entities pane, just **click the entity name**.
- In a graphics window, what you do depends on whether the selection you make is unambiguous or ambiguous:

  a) When the entity you want to edit is <u>un</u>ambiguous (the mouse target overlaps no other entities of lowest-dimensionality that can get through the active filters), the cursor will display the icon of the entity's class (e.g.) and a tooltip will display the entity type and entity name:

  Click the entity. This selects the entity and shows the properties in the Property Manager.

  b) When the entity is <u>ambiguous</u> (the mouse points at parts of more than one entity that fit the filter criteria), the cursor will display the icon:

  1 **Click the icon**. This will bring up the Which Entity dialog.

  2 **Highlight** the entity you want, then click **Select** to bring up the properties in the Property Manager.

- If there is a need to edit one object from a long list of selected objects, select the object and click to remove all but the selected object. The properties of the selected object now appear in the Property Manager for editing.

Specifying Properties and Parents in the Edit Dialog

Name. To edit the default offered for entity name, click in the name field and make your changes. Refer to “Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Name“ for details on valid entity names.

Color. Select the entity color from the drop down color palette in the dialog.

Visibility. Drop down menu to set visibility, True or False, see ”Understanding Entities - Defining Entities - Properties Common to All Entities - Visibility”.

Layer. Set the layer by choosing it from the drop down list. Use Tools>Layers to name layers, and to turn the display of individual layers on and off.

Locked. A locked entity can not be edited or moved.

Characteristic Properties. Characteristic properties are entity-specific properties such as degree for B-spline Curves, type for Arc, pitch for Helix, draft angle for Projected Snake, style for Blister.

Relabel. Available for all curves, snakes and lofted surfaces, and some other surfaces. Allows you to relabel how the parameter t is distributed along a curve or snake or u and v are distributed within a surface.

Parents. Parents of an entity, if required, may consist of:

- One or more individual parent entities (e.g. an Offset Point requires just one individual parent: Magnet, while a Projected Point requires two individual parents: Point and Mirror). In the dialog, there is an individual field for each of these.

- One or more lists of a variable number of parent entities (e.g., a list of control points for a B-spline Curve or a list of control curves for a lofted surface). In the dialog, these appear in a list box.

To specify parent entities:

Click anywhere in the parent data box. The box’s background changes to a pink and you are in parent specification mode:

- The Selection Set Manager now lists any preselected parents or *EMPTY*.

- The Available Entities Manager is constrained to list only the entities that qualify as parents.

- Any filters you had previously set are temporarily overridden.

You can select parents by clicking (and Ctrl+clicking and Ctrl+Shift+clicking if required) in the graphics display, the Available Entities Manager, or the Surface Manager. For details on how to select entities, see ”Fundamentals - Selecting Entities”.

Note: In the Entities Manager, the icons for currently-selected entities are shown with a cyan rather than a yellow background. You can right-click in whitespace in the Entity Manager and choose Expand to Selected Entities to see them. In the Available Entities Manager, any entities selected are highlighted.

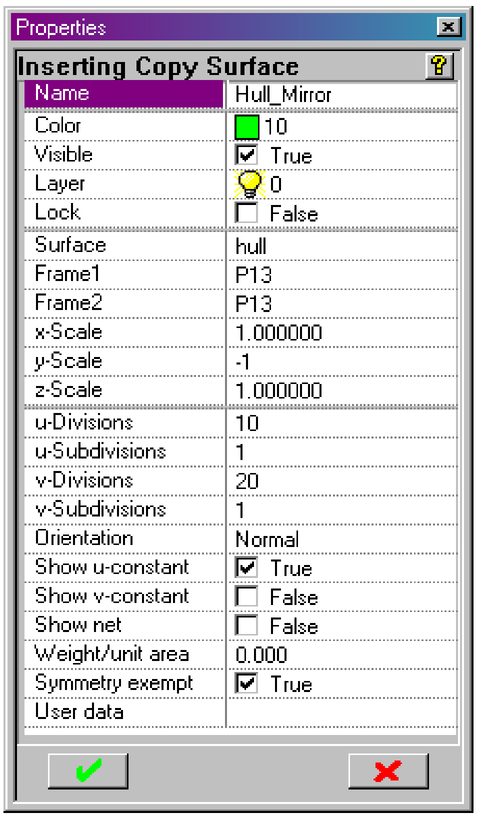
Divisions. Set divisions and subdivisions for curve and surface entities. t divisions and subdivisions are used for curves and snakes. u and v divisions and subdivisions are used for surfaces. The higher the divisions and subdivisions, the more accurately the entity will be evaluated (at the cost of processing speed). Accuracy also can be improved by coordinating divisions between parent and child entities. For further details on divisions, see "Understanding Entities - Defining Entities - Additional Curve and Snake Properties and Additional Surface Properties".

Reverse Orientation. Available for all surfaces. Reverses u and v directions.

Visibility Options. Set visibility options using drop down menus. Options available depend on the type of entity being created or edited. For explanation of the various visibility options, see "Understanding Entities - Defining Entities - Properties Common to All Entities - Visibility".

Weight. Set the weight (point), weight per unit length (curve) or weight per unit area (surface) of the entity.

User Data. User Data is a new attribute, editable in both the Property Manager and Multiple Edit. A text string of up to 40 characters can be entered here.



Using the Properties Manager Effectively

Using the Properties Manager for insertion of new entities is pretty straightforward. Select the entity from the menu, change the properties to suit the situation, and click the green check mark to complete the creation of the entity.

The process of editing existing entities showcases the functionality of the Properties Manager much better. The first thing you will notice is that a change made to most Attribute fields results in an immediate change to the entity. A minor exception to that would be fill-in attributes like Divisions or User Data. The program needs to know when the user has finished data entry. Pressing <Enter> or tabbing to the next field will complete the data entry.

In the case of the Copy Surface the parents are added, by selecting an item to have it added to replace an existing item, or by filling in a value for scale factors. When filling in a value field it is easiest to simply confirm with a press of the <Enter> button on the keyboard, but a check-mark button is provided as well.

When you select a Parent field you enter what we call “Parent Picking Mode” and can only get out of the mode by offering a confirmation. The confirmation methods are clicking the green check-mark in the field, pressing <Enter>, or tabbing to the next field. The fact that a confirmation is needed can also be a drawback to streamlined editing, so a method was implemented of querying supports without entering “Parent Picking Mode”. The parent field is actually two different fields, the name or title, and the value. If you click in the value field, you are switched to the mode of picking supports, but if your intention to just make a query of the support, simply click in the name field. If the support(s) is visible in the model they will be highlighted for your examination.

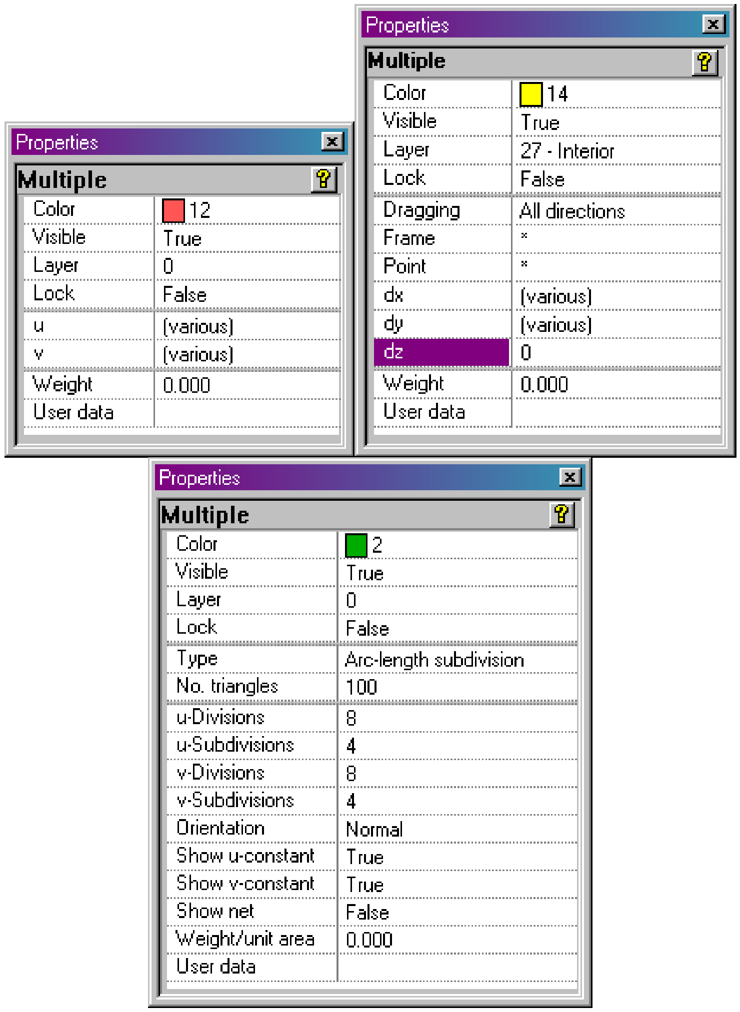
Note About Editing Point Entities that Have Relative Coordinates

Points with a Point parent and beads, magnets and rings with relative positioning: When you change the parent for any of these, or when you change from relative to absolute coordinates (or vice versa), SurfaceWorks gives you the option of keeping the absolute position of the point or keeping the offset distance the same. Additionally, for XYZ Beads and XYZ Rings, position is automatically preserved when you change the type of X, Y or Z constraint unless there are multiple solutions, in which case SurfaceWorks picks the solution closest to the t=0 end of the curve or if a bead or ring is used as a parent, SurfaceWorks picks the solution closest to the bead or ring. (Of course, you can change the initial location values should you so desire.)

Editing Multiple Selected Entities

Only the properties shared by the selected group are displayed:

In the images below we see 3 different examples of the Multiple Edit. Generally the coordinate values and / or type are shown. The first image is an edit of 2 magnets showing the ability to edit both u and / or v values at the same time. The second image shows the points involved in this Multiple Edit have all had their z coordinates changed to 0. The third image is editing multiple Trimmed Surfaces and now the types of all can be changed simultaneously.



If the indicated property currently is the same for all entities in the group, that value is shown. If the several entities have various values for the property, then “(various)” is displayed.

Note: Changes in layer, color, and visibility have no effect on children. Changes in divisions or parents may have effects on children; for example, changing the divisions of a curve or surface will affect the accuracy of its children (in most cases, increasing divisions will increase accuracy).

Layer Names and the Edit Layer Dialog

Edit a layer by choosing a number or name from the drop down list.

Note: Layer names are not editable in this dialog. If you want to name new layers or edit the names of layers, use **Tools>Layers**.

Fairing Curves

Smooth or “fair” curves in a design are often desirable, sometimes vital, both for practical and aesthetic reasons. They can add great value to objects as diverse as shampoo bottles, kitchen utensils, and automobiles — and at very low cost to the designer and builder. Especially when there is no conflict with performance objectives, it is almost criminal to design an ugly line when a pretty one would serve as well.

What constitutes a "fair" line? Simplicity is one simply stated criterion. A curve should be no more complex than it needs to be to serve its function. It should be free of unnecessary inflection points (reversals of curvature), rapid turns (local high curvature), flat spots (local low curvature), or abrupt changes of curvature (such as a straight line joining a circular arc).

All these qualities describe fairness in terms of the distribution of curvature along a curve. Consequently a tool for visualizing the distribution of curvature is a powerful aid in shaping pleasing curves. This is **View>Display>Curvature Profile**.

Curvature Profiles: Basic Reading

First, the curvature profile graph is a plot of the curvature of the selected curve vs. arc length in its current screen projection. For instance, Fig. 1 and Fig. 2 (see below) show a curve and its curvature profile in Front view and in Top view (the curve and curvature profile would vary at all rotational increments in between as well). Since the curvature profile is marked at t-increments of .1, turning on tickmarks for the curve can provide a good comparative reference. (You turn on tickmarks using the Properties Manager; in the 'Show tickmarks' box click 'True') The circle around one end of the graph axis shows which end of the graph corresponds to the t = 0 end of the selected curve.

In the graph, arc length (measured from the t=0 end of the curve) is the horizontal axis and curvature is vertical. Curvature is the inverse of the radius of curvature — a tight curve has high curvature, a straight portion has zero curvature. The sign is significant:

- positive curvature (above the graph axis) indicates the curve is curving toward the left as you move in the direction of increasing t
- negative curvature (below the graph axis) indicates the curve is curving toward the right as you move in the direction of increasing t (see Fig. 1 below)

A zero crossing of the horizontal axis in the curvature profile is an inflection point on the curve (see Fig. 2 below).

Putting yourself in "bug-mode" may be the easiest way to understand all of this. How to be a bug? Just imagine it — you are your favorite bug, walking on your screen in the direction of increasing t, along the selected curve. If as you walk you are curving to your right, that portion of the curve has negative curvature. And conversely, if as you walk you are curving to the left, that portion of the curve has positive curvature. If you are walking straight as an arrow, that portion of the curve has zero curvature (see Fig. 3 below).

Examples

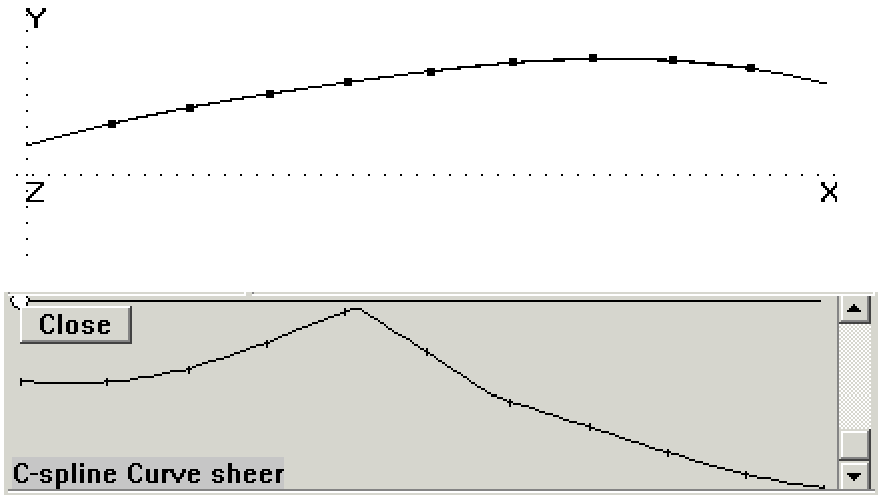


Fig. 1. Curve with tickmarks (top) and graph (bottom) in Front view.

Let's read the graph in Fig. 1. The question you might have about the curve in this view is whether or not there is an inflection between t = .3 and t = .5. The graph proves there is not — the graph does not cross the axis; the curvature is all negative (below the axis). The graph shows a slight increase in (negative) curvature at the beginning of the curve, then a gradual decrease to the area of least curvature between t = .4 and t = .5, then a gradual increase for most of the rest of the curve.

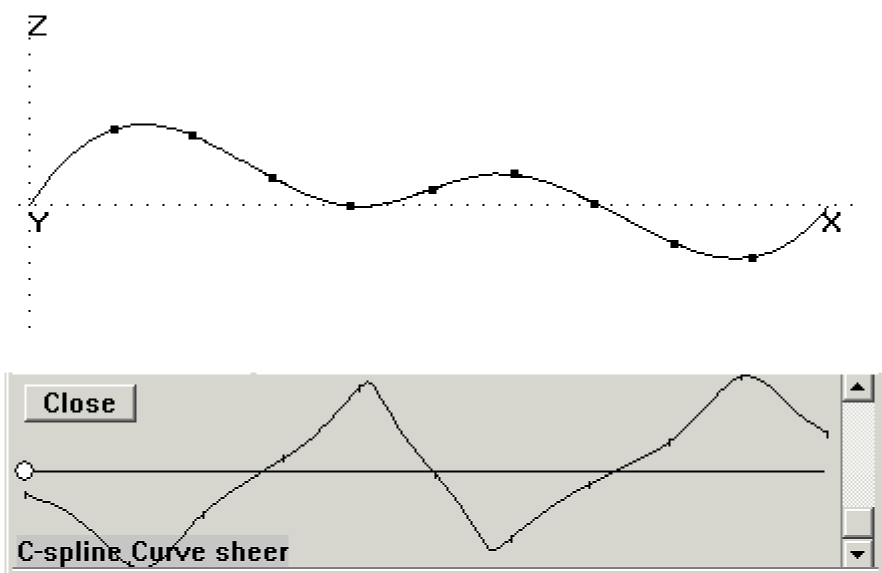


Fig. 2. Curve with tickmarks (top) and graph (bottom) in Top view. There are three inflection points in this view.

The graph in Fig. 2 is for the same curve as in Fig. 1, but in a different view. It shows three inflection points. The curvature gradually increases and decreases; there is no sharp discontinuity in the curve. By contrast, the graph in Fig. 3 (for a degree-2 B-spline Curve) shows a sharp discontinuity in curvature, at t = .5.

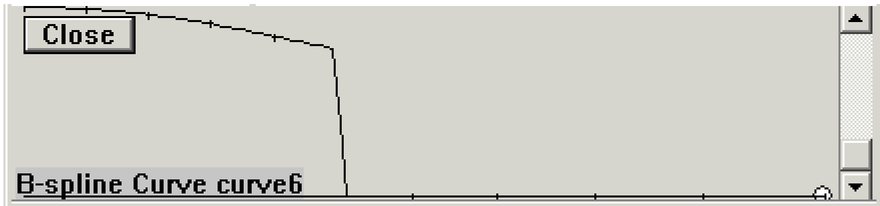


Fig. 3. This graph shows zero curvature (a straight portion on the curve) from t = 0 to about t = .5.

Continuing to read Fig. 3, note that t = 0 for this graph is at the right-hand end of the axis. The stretch of this graph with zero curvature proves that the curve is straight from t = 0 to almost t = .5, then there is the sharp increase in (positive) curvature, then the curvature is almost constant to the end.

Fairing Surfaces

Simplicity is a major factor in constructing fair surfaces. The fewer control points you use for a spline curve, and the fewer control curves you use for a lofted surface, the more automatically the surface fairs. The other side of the coin is, of course, the fewer control points and control curves you use, the less control you have over the shape of the curve or surface in between its parents.

A great deal can be done toward fairing surfaces by fairing the curves from which they are built. There also are tools for displaying the distributions of curvature measures over surfaces. These have considerable potential for revealing unfairness in surfaces.

Surface Curvatures

Surfaces, having more complexity than curves, have more ways to measure their curvature.

Definition 1: Normal Vector — At a given point on a sufficiently smooth surface, there is a unique tangent plane. The normal vector is attached to the given point and is perpendicular to the tangent plane at that point (Fig. 4, left).

Definition 2: Normal Plane — At a given point on a surface, any plane that passes through that point and includes the normal vector is called a normal plane (Fig. 4, right).

There is an infinite family of normal planes at any given point. The members of this family can be distinguished by the angle A that they make with the v = constant snake through that point.

Any of the normal planes intersects the surface. The intersection is a curve in the surface.

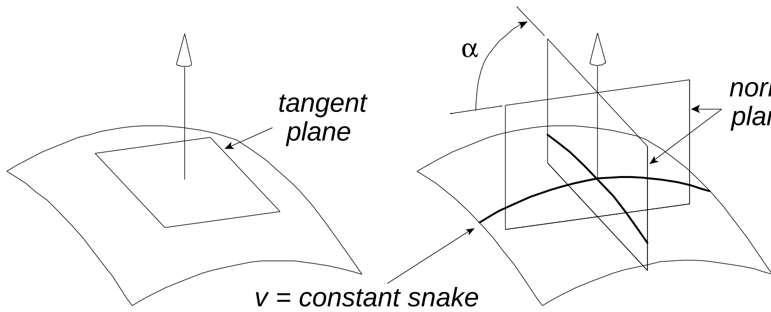


Fig. 4. Normal vector (left); normal planes and normal curvature (right).

Definition 3: Normal Curvature — Normal curvature is the curvature of the curve created by intersecting the surface with a normal plane (Fig. 4, right).

Normal curvature is a signed quantity. If the curve of intersection turns toward the positive normal direction, normal curvature is positive; if it turns away, normal curvature is negative. The units of normal curvature are 1 / length, e.g. 1 / ft if your model is in feet. The radius of curvature is the reciprocal of normal curvature. For instance, normal curvature of .03 is equivalent to that of a circle with a radius of 33.333 units (33.333 = 1 / .03).

Normal curvature depends on the orientation of the normal plane. As angle A varies, in fact, the variation of normal curvature is sinusoidal (Fig. 5), with (in general) a maximum and a minimum at angles A that are 90 degrees apart. (The exception is where normal curvature is constant with respect to A, such as any point on a sphere.)

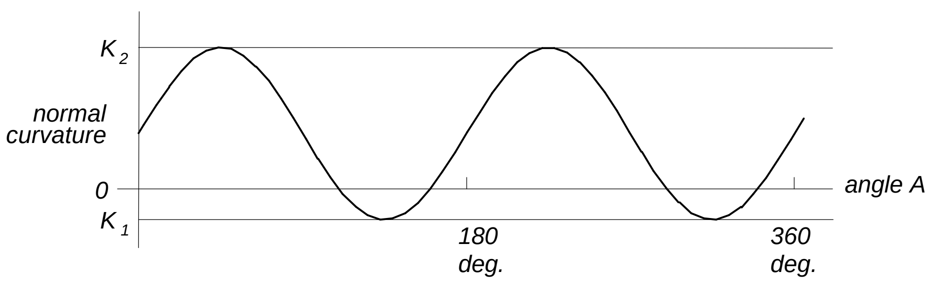


Fig. 5. As angle A is changed, the normal curvature varies between the limits of the two principal curvatures.

Definition 4: Principal Curvatures — The maximum and minimum values of the normal curvature at a point are called the principal curvatures K_1, K_2 of the surface at that point.

The units of principal curvature are 1 / length, e.g. 1 / ft if your model is in feet.

Definition 5: Mean Curvature — The average of the two principal curvatures, $(K_1+K_2) / 2$, is called the mean curvature of the surface at that point.

The units of mean curvature are also 1 / length.

Definition 6: Gaussian Curvature — The product of the two principal curvatures, $K_1 \times K_2$, is called Gaussian curvature.

Because K_1 and K_2 are signed quantities, so is Gaussian curvature. At a point on the surface where the principal curvatures have the same sign, Gaussian curvature is positive (Fig. 6, top). This type of surface is locally shaped like an ellipsoid or a sphere. At a point on the surface where the principal curvatures have opposite signs, the Gaussian curvature is negative (Fig. 6, bottom). This type of surface is locally shaped like a saddle or a potato chip. If one principal curvature is zero, the Gaussian curvature is also zero (Fig. 6, middle). If the Gaussian curvature is zero everywhere, the surface is developable, i.e. it can be rolled out flat onto a plane without stretching. Nonzero Gaussian curvature is often loosely referred to as compound curvature.

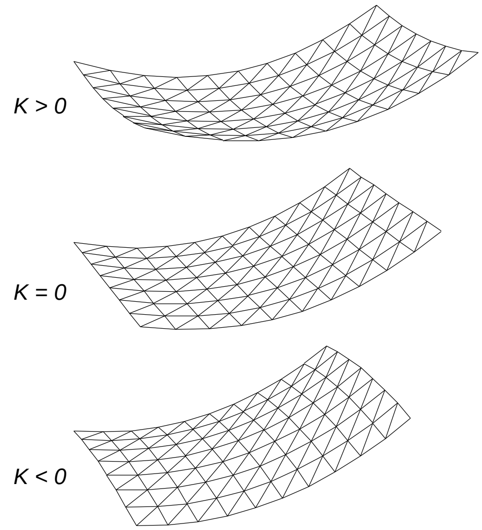


Fig. 6. Positive (sphere-like), zero (curved sheet), and negative (saddle-like) Gaussian curvature.

The units of Gaussian curvature are 1 / (length squared), e.g. 1 / ft^2 if the model is in feet.

Note: The principal, mean, and Gaussian curvatures are properties of the surface at a point. They are independent of any coordinates chosen to describe the surface and also of any snakes in the surface that happen to pass through the point. You can remove the coordinates, and remove any snakes in the surface, and these curvature properties remain unchanged. (The color displayed and the contours are independent of your choice of angle.)

By contrast, the normal curvature at a point depends on the direction (angle) of the normal plane passing through the point.

Some Properties and Applications for Surface Curvatures

- Smooth, fair surfaces generally show gentle color gradients for all these curvature measures. Local small bumps or hollows show up as abrupt color changes.

- Normal curvature is something like rocking a straightedge on the surface, along the direction of the selected line snakes. A zero-curvature place along such a line would be a place where the straightedge would bump or click because of a local flat spot or inflection.

- Negative Gaussian curvature indicates that there are hollows in some directions at that place on the surface.

- Gaussian curvature of zero is required for a ruled surface (Ruled Surface or B-spline Lofted surface with two control curves) to be developable. The amount of Gaussian curvature quantifies how far a compound-curved surface is from developable, i.e. difficulty of forming from flat material.

Relabeling Curves and Snakes

Introduction

To begin with, we'd like to emphasize that relabeling is a specialized tool which is not needed in the great majority of models. In a recent survey of over 600 models we found fewer than 3% of the curves and snakes and fewer than 1% of the surfaces used non-default labeling. On the other hand, when you do need it, relabeling easily achieves geometric effects that would be difficult or impossible by any other means within SurfaceWorks.

As we have noted elsewhere, all curve and snake class entities (we'll just use the general term "curve" in this section to cover both) are labeled with a parameter t which varies from 0 at the start of the curve to 1 at its end. The distribution of the parameter t along the curve is called the "labeling" or "parameterization" of the curve. Each curve has a "natural" labeling, which is what you get by specifying the default '*' for the curve's Relabel entity. For special purposes, such as controlling the shape of lofted surfaces or controlling the size distribution of panels generated from a surface, it is sometimes necessary to have a way to control the labeling, and vary it from the default. This is the function of the SubCurve/SubSnake (see "SubCurve and SubSnake Alternatives" on page 92) and the Relabel entity (see "The Relabel Entity" on page 92).

Similarly, all lofted surface entities have a natural labeling of their lofting curves (the v direction on the surface). By use of a SubCurve/SubSnake or a non-default Relabel entity, the distribution of v and the size distribution of panels in the v-direction can be controlled in a very flexible fashion.

Unlike most SurfaceWorks entities, Relabels are not visible entities themselves; however, they have geometric effects on the visible entities they modify, that usually produce visible changes.

What Does and Does Not Change

It is important to realize that:

- Relabeling does NOT change the set of points that make up the actual continuous curve or surface. It only changes the way these points are labeled with parameter values.
- Relabeling CANNOT be used to reverse the direction of the t or v parameters of a curve or surface — depending on the kind of curve/surface, you would reverse its orientation by reordering its control points/curves (e.g. B-spline Curve, Blend Surface), using the "Reverse snake orientation" check box (e.g. Edge Snake), etc.; or you might create a SubCurve/Snake.
- Relabeling DOES affect the display of curves as polylines, and surfaces as meshes. It can make the individual polyline or mesh segments shorter or longer, and more or less visible, in different parts of the curve or surface.
- Relabeling DOES cause the location of tickmarks displayed on a curve or v=constant parametric lines on a surface to shift to new positions.

- Relabeling parent curves of a surface DOES affect (often dramatically) the mesh of coordinate lines used to display a surface, and it often has an effect (usually subtle) on the shape of the surface itself.

Default Labeling

Unless you specify otherwise by using a non-default Relabel entity, SurfaceWorks uses the “natural” distribution of t along any curve. The natural distribution is specified in the description of each curve, snake or surface entity; for example:

- For a line or arc, the natural distribution of t is uniform with respect to arc length; i.e., uniformly spaced t-values divide the curve into intervals of uniform length.

- For a B-spline, the natural distribution of t is strongly influenced by the spacing of control points, i.e., concentrated in areas where control points are close together. This helps to automatically provide improved resolution in highly curved areas.

- For a C-spline, the natural distribution of t is approximately uniform with respect to arc length.

For each kind of curve, the natural distribution is simply the way the parameter t most “naturally” fits into the equations that define the curve. Since each curve entity is defined by its own set of equations, what is natural for one kind of curve has nothing to do with what is natural for another kind.

SubCurve and SubSnake Alternatives

Note that the SubCurve and SubSnake entities provide alternative ways to create a relabeled curve / snake — with interactive, visual control by dragging beads. There is an exact correspondence in the resulting labelings. For example, a degree 2 SubCurve (or SubSnake) with five beads at t = 0, t = 0 t = 0.5, t = 1, and t = 1 produces the same labeling as a degree 2 Relabel with parameter values [0.   0.   0.5   1.0   1.0]. For details, see SubCurve and SubSnake in the Entity Descriptions.

The Relabel Entity

A Relabel entity is really a simple form of a B-spline curve, whose “control points” are a series of t parameter values — numbers, rather than points. The graph of the Relabel curve is the key to understanding and controlling relabeling.

Figure 7 is an example of the Relabel graph, created by a degree 2 relabel entity with the three parameter values [0.   0.8   1.0].

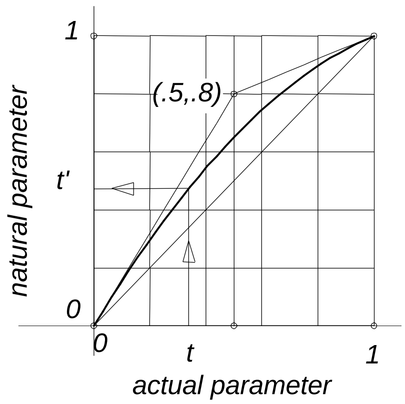


Fig. 7 Relabel graph for type-2 Relabel entity with 3 parameter values [0. 0.8 1.0].

The vertical axis of the graph is the “natural” parameter value t’. The horizontal axis is the actual parameter t. A point on the curve at parameter value t is located by:

1. entering this graph with t and getting back a natural parameter t’.
2. evaluating the curve at natural parameter t’.

Here is the equation for the t / t’ relationship we are graphing:

$$t' = \sum_{i=1}^{n} t_i \, B_i^k (t)$$

where

n = number of parameter values
t_i = parameter values i = 1, ..., n
k = B-spline order = type + 1

Figure 8 left shows the effect of the above example Relabel on a simple curve, an Arc, drawn with 40 t-divisions. In this case, the number of divisions is large enough that without the tickmarks, you won’t see any difference between the displays of the natural and relabeled curves.

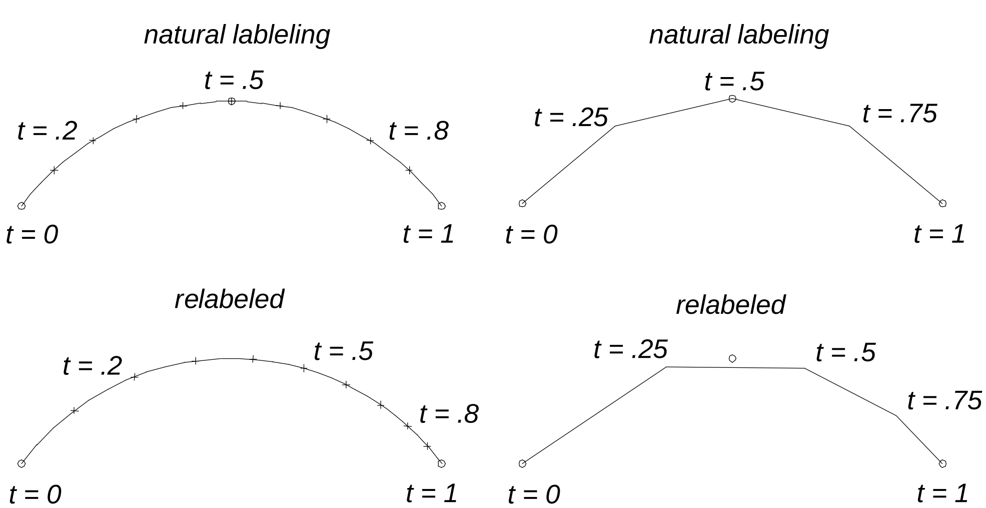


Fig. 8 (left) and (right) Natural and relabeled t distributions — arc with 40x1 divisions (left); with 4x1 divisions (right).

In Figure 8 right, however, with only four t-divisions, the polyline representation of the arc is noticeably different following relabeling. (The points plotted for a polyline with t-divisions = 4 are at t-values of 0, .25, .50, .75, 1.)

A Relabel entity can be of any B-spline type (integer > 0) and can have any number of parameter values (>1). The parameter list should start with 0. and end with 1.; SurfaceWorks will substitute these values in any case. It is required that t' be a monotone non-decreasing function of t (i.e., the curve of t' vs. t would not have any parts that slope downward). Additionally, it is required that t' not have 0 slope for any finite intervals of t. Note that this does not disqualify the slope of t' from being 0 for a single value of t.

To test whether these requirements have been met, SurfaceWorks examines the parameter values. If the parameter values are strictly increasing both requirements are automatically satisfied and the relabel is valid. For example:

[0. .25 0.5 .75 1.]
and
[0. 0.6 1.]

are both increasing sequences.

If the parameter values are not strictly increasing, but are still non-decreasing (when 1 or more parameter values are repeated, for example), the second requirement may or may not be met. For example:

[0. 0. 0.5 1. 1.]
and
[0. 0.6 0.6 1.]

are both non-decreasing sequences. In these cases, however, SurfaceWorks examines the parameter values to determine whether any value is repeated more than 'Degree' times, 'Degree' being the Degree of the Relabel. If one or more values fail this test, t' has 0 slope for a finite interval and the second requirement is not met. This situation results in the relabel being placed in error as does the situation where the parameter values are not a non-decreasing sequence.

Note: It is possible for the B-spline function (t') that is produced from a set of Relabel parameter values to be valid even if the parameter values fail the tests described above. Because SurfaceWorks examines only the parameter values, such cases still result in the Relabel being placed in error. Also for this reason, and because the monotonicity requirements are a recent addition to SurfaceWorks, Relabels in models from old .ms2 files may experience errors.

Figure 9 is a graphical look at a relabel that has non-decreasing parameterrs:

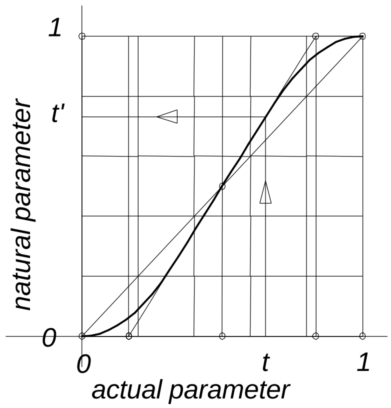


Fig. 9 An example of a non-decreasing set of parameters [0. 0. 0.5 1. 1.].

The following set of parameter values is not non-decreasing (Fig. 10):

[0. 0.6 0.4 1.0]

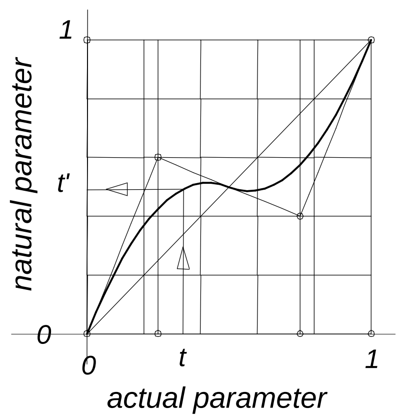


Fig. 10 A “parameter loop” created by a NOT non-decreasing set of parameter values [0. 0.6 0.4 1.].

In a type-1 or type-2 Relabel, this set of values would try to create a “parameter loop” in which part of the curve (near the middle in this case) is traversed 3 times as t goes from 0 to 1. This will cause an error in SurfaceWorks.

“Natural” Relabel Entities

If the Relabel graph is just t’ = t (a straight line from 0,0 to 1,1), the labeling is going to come out natural; this particular relabeling has no effect. There are many combinations of type and parameter values that give this result. We call such a Relabel entity “natural”. Of course, if what you want is natural labeling, it’s a waste of time to make a natural Relabel — just use ‘*’ instead. But just as it’s convenient to start your car in neutral, then shift gears to go forward, a natural Relabel is often a good starting point for relabeling a curve: first, create a natural Relabel that has no effect; apply it to the curve that needs relabeling; then adjust its parameter values to achieve the desired relabeling effect.

Here are parameter values that create natural Relabels (nv = number of values):

type	nv	parameter values					
1	2	0.	1.0				

1	3	0.	0.5	1.0			
1	4	0.	0.3333	0.6667	1.0		
1	5	0.	0.25	0.5	0.75	1.0	
1	6	0.	0.2	0.4	0.6	0.8	1.0
1	etc.						

type	nv	parameter values							
2	2	0.	1.0						
2	3	0.	0.5	1.0					
2	4	0.	0.25	0.75	1.0				
2	5	0.	0.1667	0.5	0.8333	1.0			
2	6	0.	0.125	0.375	0.625	0.875	1.0		
2	7	0.	0.1	0.3	0.5	0.7	0.9	1.0	
2	8	0.	0.0833	0.25	0.4167	0.5833	0.75	0.9167	1.0
2	etc.								

type	nv	parameter values							
3	2	0.	1.0						
3	3	0.	0.5	1.0					
3	4	0.	0.3333	0.6667	1.0				
3	5	0.	0.1667	0.5	0.8333	1.0			
3	6	0.	0.1111	0.3333	0.6667	0.8889	1.0		
3	7	0.	0.0833	0.25	0.5	0.75	0.9167	1.0	
3	8	0.	0.0667	0.2	0.4	0.6	0.8	0.9333	1.0
3	etc.								

The interval patterns are:

1 : 1 : 1 ... 1 : 1 : 1 for type-1
1 : 2 : 2 ... 2 : 2 : 1 for type-2
1 : 2 : 3 : 3 ... 3 : 3 : 2 : 1 for type-3.

These patterns are visible when you look at the fractions in the following two tables:

type	nv	parameter values						
2	2	0	1					
2	3	0	1 / 2	1				
2	4	0	1 / 4	3 / 4	1			
2	5	0	1 / 6	3 / 6	5 / 6	1		
2	6	0	1 / 8	3 / 8	5 / 8	7 / 8	1	
2	7	0	1 / 10	3 / 10	5 / 10	7 / 10	9 / 10	1

type	nv	parameter values						
3	2	0	1					
3	3	0	1 / 2	1				
3	4	0	1 / 3	2 / 3	1			
3	5	0	1 / 6	3 / 6	5 / 6	1		
3	6	0	1 / 9	3 / 9	6 / 9	8 / 9	1	
3	7	0	1 / 12	3 / 12	6 / 12	9 / 12	11 / 12	1

Example Parameter Values for Relabels

If you are going to change the distribution of t from the default distribution, you'll need to specify an appropriate set of parameter values for your new Relabel entity. So, let's look at how different sets of parameter values affect the distribution of t on a specific curve — a Degree-2 B-spline Curve. Here are several possible 5-value, type-2 Relabels:

Relabel R1 / 2 {0. .1667 .5 .8333 1. 0} is a "natural" Relabel

Relabel R2 / 2 {0. .45 .5 .55 1 } packs t-values into the middle of the curve

Relabel R3 / 2 {0. .5 .8 .9 1} packs t-values at the t=1 end

Relabel R4 / 2 {0. 0. .5 1. 1.} packs t-values at both ends of the curve

In all these cases, the curve itself consists of the same set of points. What's different is how the points are labeled with t-values, and this affects:

- the display of the curve with a finite number of points
- the shape of a surface built from the curve
- the panelization of a surface built from the curve

Here's how the DEMO hull, which has a type-2 B-spline Curve as its second master curve (MC2), would vary with use of this set of Relabels. In Figure 11 we apply these relabels, in order, to MC2 (not to the other master curves, or to the C-spline Lofted Surface 'hull'). Note particularly the sweep of the u=constant lines (lofting C-splines) that join points of equal t-value on each of the three master curves. Since a C-spline Lofted Surface interpolates all of its master curves, the various surfaces agree exactly at 'MC1', 'MC2' and 'MC3'; but in between the master curves, the surfaces differ.

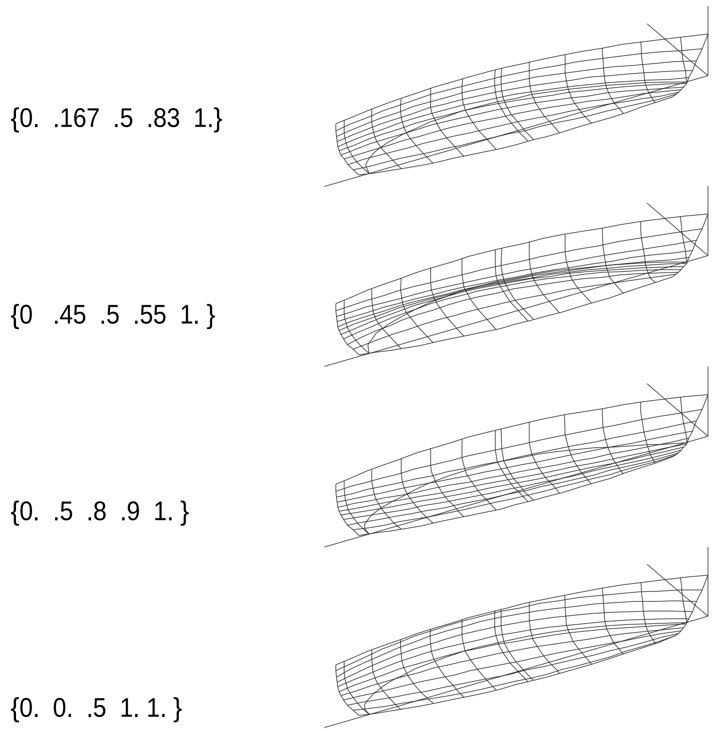


Fig. 11

In Figure 12 we apply the same Relabel 'R4' to all three of the master curves. This creates the exact same surface as the original model DEMO.MS2, but with a different distribution of the u=constant parameter lines.

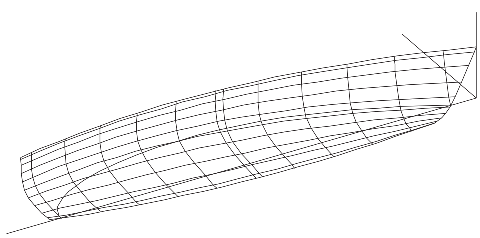


Fig. 12

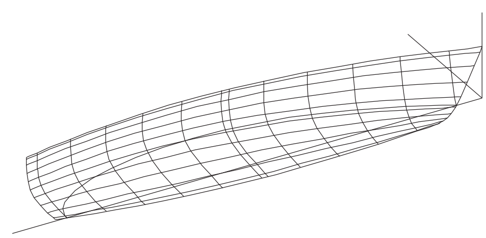


Fig. 13

In Figure 13 we also apply Relabel 'R4' to the C-spline Lofted Surface 'hull'. Again, the surface is identical to the original DEMO model, but the distribution of both u=constant and v=constant lines is dramatically shifted toward the edges.

Finally, Figure 14 shows an example of a more dramatic or qualitative shape change achieved with relabeling. ‘worm’ is a Sweep Surface which uses a graph ‘h1’ to taper the radius down from the middle toward both ends. However, since a BGraph has finite slope at the ends, the surface comes out pointed rather than rounded at nose and tail. This is corrected by applying Relabel ‘R4’ (above) to the ‘path’ curve. Packing t-values toward both ends of ‘path’ allows both ends of the Sweep Surface to be smoothly rounded.

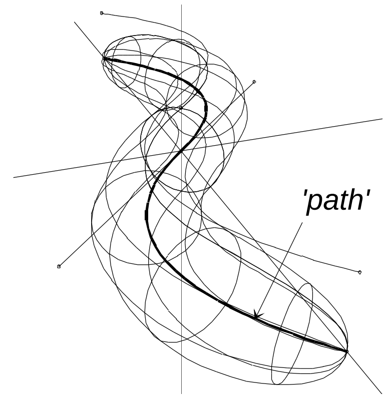


Fig. 14

Understanding SurfaceWorks Entities

Introduction

In this , we explain the classification of SurfaceWorks entities, provide a list of the entities, by class, and detail the properties various entities share in common. (In the “Entity Descriptions” chapter we detail the properties and parents unique to each entity definition.)

Classification of Entities

SurfaceWorks entities fall into a few neat groups according to their dimensionality. The SurfaceWorks entity classes are:

Points (zero-dimensional)
- 3D Points
- Beads
- Magnets
- Rings
- TriMesh Magnets
- TriMesh Rings

Curves (one-dimensional)
- 3D Curves
- Lines
- Snakes
- TriMesh Snakes

Surfaces (two-dimensional)

Solids (three dimensional)

Other

See a complete list of entities at the end of the Entity Descriptions chapter.

Defining Entities

Overview

We use the term “entity” to refer to a kind of geometric entity. To create an instance of an entity, you:

1 Select the kind of entity it is to be.

2 Specify the particular properties and parents (if any), for that individual entity.

For example, to create a Projected Point entity, you would select

Insert>Point>Projected Point (or  >**Projected Point**), and then specify the following properties and / or parents (or accept the defaults offered):

- name
- basis point (the Projected Point's parent)
- mirror
- layer
- color
- visibility
- locked status
- weight

In this chapter, we explain the properties common to most entities or entity classes (points, curves, etc.).

Summary of Property Data

All kinds of entities need at least the following property data:

name
layer
color (except for Entity List and Relabel)
visibility (except for Entity List and Relabel)

All curves and snakes need the following property data:

t-divisions
t-subdivisions

All surfaces need the following property data:

u-divisions
u-subdivisions
v-divisions
v-subdivisions

Beyond these properties, the balance of each entity's definition varies.

Properties Common to All Entities

Name

ALL entities have a name property. An entity name labels a specific entity. For instance, 'top' might be the name of a Point located at x=0, y=0, z=10. Entity names may be used in the creation of subsequent entities. Entity names are user-defined and can be up to 16 characters. Duplicate entity names are not allowed within a model. You can use any combination of letters and numbers, but most punctuation marks and symbols are illegal. The following are not allowed in entity names:

space, semicolon (;), colon (:), comma (,), period (.), parentheses ( ), slash (/), asterisk (*), question mark (?), double quote mark ("), braces ({ }), plus sign (+), minus sign or hyphen (-), caret (^)

The underscore character '_' can conveniently be used where you would like a space or a hyphen; for example, 'wild_side'.

Note: SurfaceWorks will insert this underscore automatically if you type a space in the name.

Entity names ARE case sensitive. If you so chose, you could use ‘ABC’, ‘abc’, and ‘AbC’ to name three different entities. Under no conditions would the program recognize these three names as referring to the same entity.

Sometimes we like to use names that tell what physical entity the geometric entity represents; sometimes we prefer, especially for points, short names in a simple sequence. When speaking of entities, we often use the kind of entity and entity’s name together to identify an individual entity, for example “Point pt3” or “C-spline Curve top_outline”. The kind of entity (e.g. Point or C-spline Curve) is like a family name, and the name property (e.g. ‘p3’ or ‘top_outline’) is like a given name.

Layer

All entities have a layer property, meaning that they are assigned to a specific layer. An entity can be on only one layer at a time. Layers are identified by number (an integer 0-255). In addition, layers may be named (using **Tools>Layers**).

The default layer assignment when an entity is created is the Current Layer (set using **Tools>Layers**). The first time you open a model in SurfaceWorks, the default Current Layer is Layer 0 (Layer 1 in SolidWorks Integration mode).

Color

All entities except Entity List and Relabel have a color property that you set by clicking a color swatch. Sixteen colors are available:

Black	dark gray
Dark blue	blue
Dark green	green
Dark cyan	cyan
Dark red	red
Dark magenta	magenta
Brown	yellow
Gray	white

Note: Depending on your computer system and the brightness of your screen, the colors may appear somewhat differently (e.g. red may appear pink or brown).

Visibility

All entities except Entity List, Formula, Variable, Graph, Knotlist and Relabel have a visibility property. Some visibility options vary according to the class of entity. The one option all share is that any entity can be hidden (or not hidden, of course!).

Additional Visibility Options

For curves and snakes, you can display:

- tickmarks along the curve / snake at intervals of 0.1 in the parameter t
- the polyline connecting the control points (for Arcs, B-spline Curves, C-spline Curves and Foil Curves)

For surfaces, if **Tools>Options>Display>Display mesh lines** is checked you can display:

- u-constant lines
- v-constant lines
- boundary lines
- Fitted Surfaces can display 'Net' and Trimmed Surfaces can display 'Triangles'

Note: For surfaces, turning off u- and v-constant lines and boundary lines is the same as hiding the surface (i.e. it will not display).

The visibility options can be used singly or in combination.

Additional Curve and Snake Properties

Divisions: t-divisions and t-subdivisions

All curves and snakes have t-divisions and t-subdivisions properties. The number of t-divisions x t-subdivisions specifies the number of line segments SurfaceWorks will use to approximate a curve (or snake) for display and internal tabulation. t-divisions and t-subdivisions are both integers (1-255). The greater the total number of divisions (divisions x subdivisions), the more closely the curve screen display will approximate the curve itself (which is composed of an infinite number of points).

Let's look at the displays of two B-spline curves whose definitions differ only in the number of divisions. The first, specified with divisions = 5x1, is displayed with 5 line segments (Fig. 1, left); the second, specified with divisions = 40x1, is displayed with 40 line segments (Fig. 1, right). Note that divisions = 5x8 would look exactly the same as 40x1.

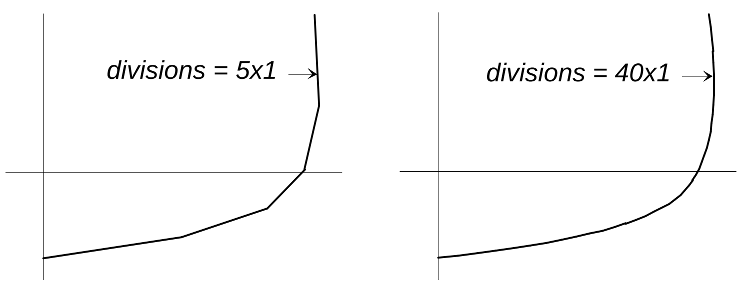


Fig. 1. A curve drawn with 5x1 t-divisions (left) and 40x1 t-divisions (right).

The main reason you might want to use more than one t-subdivision for a curve or snake is to have more than 255 divisions. For instance, if your entity is a 10-turn helix (angle = 3600 degrees), 255x1 divisions might not be enough for adequate accuracy since the calculations use the polyline in place of the actual curve — each turn would be represented by only 25.5 segments.

For details about the parameter t, see "Basic Concepts - Parametric Curves"

relabeling

All curves and snakes have a relabel attribute, used to distribute the parameter t along the curve (or snake). The default relabel, '*', produces the "natural" labeling for the curve. This natural distribution of the parameter t along a particular kind of curve is described in the Entity Description for that curve (or snake).

In most instances, the default labeling is quite satisfactory. If you choose to relabel a curve, you must create a Relabel entity that specifies the new labeling and include the Relabel entity's name in the curve's definition.

Note that a Relabel entity CANNOT be used to reverse the t-orientation of a curve — depending on the kind of curve, you would do that by reordering its control points (e.g. B-spline Curve), using the "Negative orientation" choice(e.g. Edge Snake), etc; or you might create a SubCurve / Snake.

Additional Surface Properties

Divisions: u and v divisions and subdivisions

All surfaces have u-divisions, u-subdivisions, v-divisions, and v-subdivisions properties. The number of u (or v) divisions x subdivisions = the number of line segments SurfaceWorks uses to represent the surface in the u (or v) direction for display and internal tabulation. The product of all divisions and sub-divisions cannot exceed 160,000. u-divisions, u-subdivisions, v-divisions, and v-subdivisions are all integers (1-255 generally speaking). With some exceptions, the greater the total number of u and / or v divisions, the more closely the surface screen display will approximate the surface itself (which is composed of an infinite number of points). Refer to the general information on divisions and subdivisions for more on this.

In addition, the number of u-divisions (or v-divisions) establishes the spacing of the u-constant (or v-constant) lines drawn in a graphics display with mesh lines turned on.

For instance, if a surface is defined with 10x3 20x1 divisions, SurfaceWorks will tabulate the surface internally with 30 segments in the u-direction and 20 segments in the v-direction. In the display, if the u-constant lines are visible, there will be 11 of them (0, .1, .2 ... .9, 1.0; Fig. 2, top). If the v-constant lines are visible, there will be 21 of them (0, .05, .10, ... .95, 1.0; Fig. 2, bottom). For details about the parameters u and v, see "Basic Concepts - Parametric Surfaces."

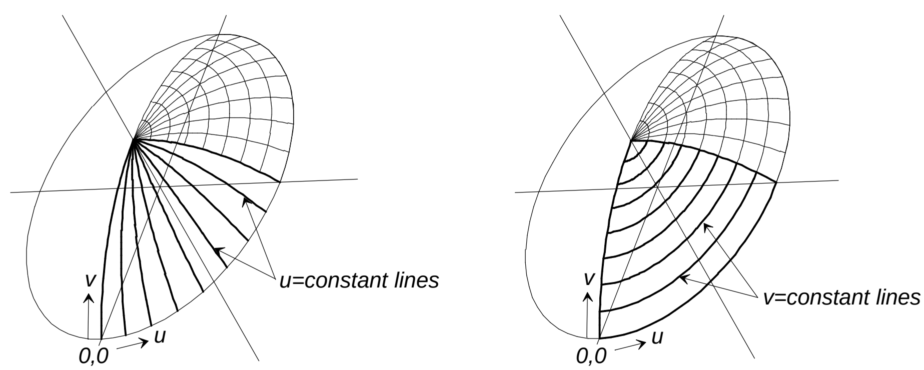


Fig. 2. u and v divisions on a surface.

Coordinating Divisions

It is important to get into the habit of coordinating the divisions of surfaces with the divisions of their parent curves. This means making parent curve divisions the same as, or a multiple of, surface divisions. In general, coordination of divisions:

- Produces a more accurate IGES export model.

- Decreases IGES export time
- In SolidWorks Integration mode, produces better quality surfaces in SolidWorks.

To coordinate divisions:

When creating a lofted surface, use the same number of divisions on each of the master curves, and also make sure that the master curve t-divisions x t-subdivisions is the same as, or a multiple of, the surface u-divisions x u-subdivisions.

E.g. to make curve and surface divisions the same, you might use:

t-divisions = 8x4 and **u**-divisions = 8x4
t-divisions = 8x4 and **u**-divisions = 4x8

E.g. to make curve divisions a multiple of surface divisions, you might use:

t-divisions = 8x4 and **u**-divisions = 4x2
t-divisions = 8x4 and **u**-divisions = 8x2

Similarly, for a Revolution Surface: make the divisions of the meridian curve the same as, or a multiple of, the surface **u**-divisions.

When creating a boundary surface or a SubSurface, make the divisions of the first and third bounding curves (snakes) the same as, or a multiple of, the surface **u**-divisions. And make the divisions of the second and fourth bounding curves (snakes) the same as, or a multiple of, the surface **v**-divisions.

Similarly, for a Translation Surface: make the divisions of the stationary curve the same as, or a multiple of, the surface **u**-divisions, and the divisions of the sliding curve the same as, or a multiple of, the surface **v**-divisions.

Note: Coordinating divisions does not mean that the number of **u**-divisions has to be the same as the number of **v**-divisions.

The Variable Portion of an Entity Definition

Beyond keyword, name, color, visibility, layer, and divisions (if any), the balance of each entity definition is the specific data properties and / or parents required to specify that entity type. For instance, all the data required to complete a Point entity with default Parents are the values of its three coordinates dx, dy, dz. The rest of the data for a C-spline Curve is the value for degree and the list of Control Points the curve goes through. The rest of the data for a C-spline Lofted Surface is the value for degree and the list of Control Curves the surface is sprung over.

For details, see the particular entity in “Entity Descriptions”.

Collective Entity Information

Most of this section is dedicated to explaining the “basics” about several groups of entities. For instance C-spline Curves, C-spline Snakes, and C-lofted Surfaces all have the same set of relationships to their control points and the same degree property; similarly for B-spline Curves, B-spline Snakes, and B-spline Lofted Surfaces. We have grouped that shared information here in “Collective Entity Information.” In the “Entity Descriptions”, in which we detail the parents and unique or “characteristic” properties of each entity, we’ve put cross-references to the collective information.

B-spline Entities

Reference for B-spline Curve, B-spline Snake, B-spline Lofted Surface entities; SubCurve and SubSnake entities made with at least 3 parents

Definition

A B-spline Curve is a continuous curve defined by a series of control points. The curve is formed in relation to the 3D polyline (i.e., the broken line) joining the points in sequence. The B-spline Curve always starts at the first control point (point1) and ends at the last control point (pointN). It is always tangent to the polyline at these end points, but in general it does not pass through the other control points (unless the curve is degree-1) see Fig 3.see Fig 3

A B-spline Lofted Surface is the same sort of entity except that it uses control curves instead of control points to define its shape.

Degrees

A degree-1 B-spline Curve is the polyline itself (Fig , left); this is one way to make a polyline entity in SurfaceWorks. For degree-2 and higher, the curve imitates the shape of the polyline, or is guided by it, but in general does not pass through the control points other than the first and last (Fig 3, center and right). The higher the degree, the “stiffer” the curve behaves, i.e. it more loosely imitates the polyline and has a higher number of continuous derivatives

Fig. 3. Three degrees of B-spline curve.

A degree-2 B-spline Curve is made of one or more parabolic arcs that join with continuous slope (except at doubled control points). In addition to being tangent to the polyline at the endpoints, the degree-2 B-spline is also tangent to the midpoint of each polyline segment other than the first and last (Fig 4, top). If three control points lie on a straight line, a portion of the B-spline will be perfectly straight (Fig 4, bottom). If two control points coincide, the curve will pass through them; the tangent as well as the curvature can be discontinuous at this point (Fig 4, middle).

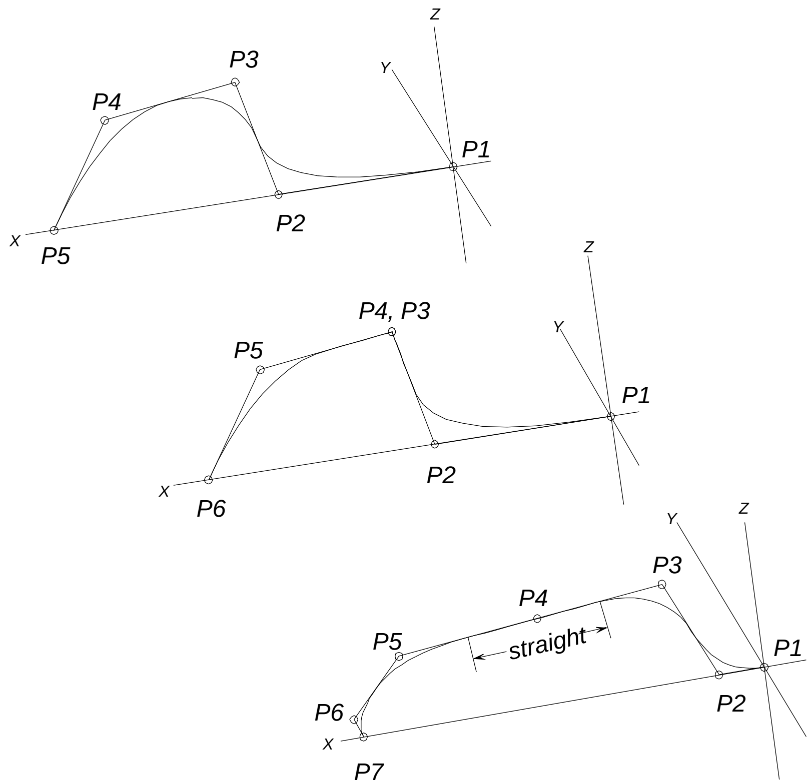


Fig. 4. Degree-2 B-spline curves: curve is tangent to polyline at endpoints and at midpoint of polyline segments other than first and last (top); doubled control points ‘P3’ and ‘P4’ allow discontinuity in curvature (middle); 3 control points in straight line make a portion of the curve straight (bottom).

A degree-3 B-spline Curve is a smooth curve made of one or more parametric cubic segments that join with continuous slope and curvature (except at multiple identical control points). While it is tangent to the polyline at the endpoints, it is not usually tangent at the midpoints of the polyline segments (Fig 3, right). If 4 control points lie on a line, a portion of the degree-3 B-spline will be perfectly straight.

Technically, a B-spline should have at least one more control point than its degree, e.g. at least 2 for degree-1, 3 for degree-2, etc. However, SurfaceWorks will automatically “demote” a spline to lower degree as necessary to fit the number of control points. For example, a degree-3 spline specified with only 2 points will be treated as degree-1; with 3 points it will be treated as degree-2; only with 4 or more points will it actually be a cubic spline.

The program upper limit for B-spline degree is 65535, though since this would require 65536 control points to achieve, it is unlikely ever to be used. In practice, we find that we rarely exceed degree 3.

A “Bezier curve” is a B-spline curve whose degree is one less than the number of control points — for example, the degree-3 B-spline in Fig 3.

Contours Entities

Reference for Contours entity.

Introduction

Contours includes contours that are parallel to a plane and those that are normal to a curve. Type 'Offset from Mirror / Surface' contours are parallel to a plane and include the "standard" contours you specify by picking one of the default planes as the Mirror / Surface, and contours you build using any other plane or surface. Type 'Normal to Curve' contours are normal to a curve and are called cvcontours in SurfaceWorks.

Definition

Contours entities create one or more parallel sections in a user specified arbitrary orientation. Each section cuts all the surfaces listed in the Contours edit dialog in the "Surfaces to cut" box, creating 3D polylines having the color and visibility designated. You specify the indices of the first and last contours ("Index of first contour" and "Index of last contour"), the offset of the 0-index contour from the basis surface ("Signed distance from Mirror/Surface to 0-index contour" or "Parameter value of 0-index contour"), and the spacing between the contours (Contour spacing).

Contours sets

For the purposes of explaining contours indices and how to specify a set of contours, we’ll use a Contours entity type "Offset from Mirror / Surface" that uses the *X=0 default plane

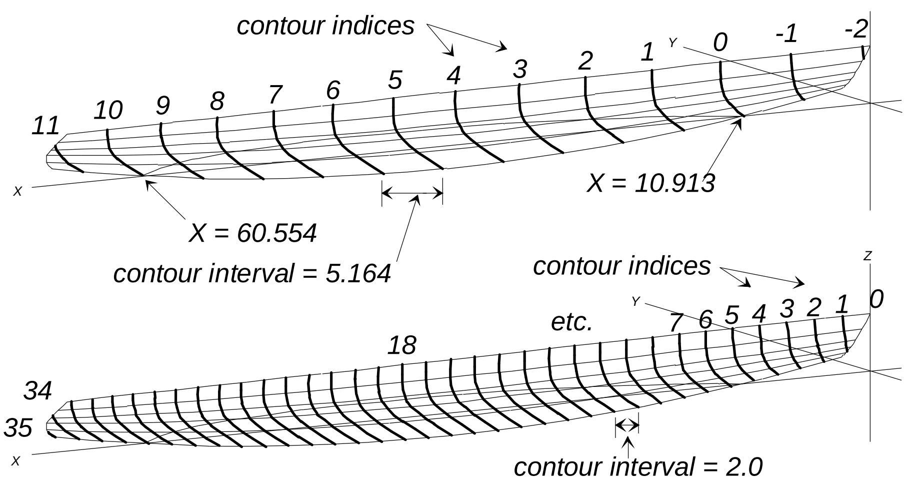


Fig. 5 A hull and two different sets of X-Contours.

Each contour has both an X-position (its X-coordinate) and an index (its sequential number in the set of contours). Figure 5 shows two possible sets of X-Contours for a hull and a set of contour indices for each of them. In the first case, we established the contour spacing by dividing the waterline into 10 equal divisions, and we put the zero-index contour at the forward end of the waterline. In the second case, we decided on a 2-ft. contour interval and put the zero-index contour at the point of the stem (X = 0).

Here’s where Index of first contour and Index of last contour come in — they are index numbers you use to specify the set of contours to include in a contour entity. In the first case above, to make the set of contours that encompasses the full length of the hull, we would specify Index of first contour = -2 and Index of last contour = 11 (and Signed distance from Mirror/Surface to 0-index contour = 10.913, Contour spacing = 5.1641 — details below). In the second case, let’s say we just want to make the set of contours that encompasses the aft half of the hull — we would specify Index of first contour = 18 and Index of last contour = 35 (and Distance from Mirror/Surface to 0-index contour = 0, Contour spacing = 2 — details below).

X-coordinates for the contours are calculated from the equation:

X = Signed distance from Mirror/Surface to 0-index contour + Contour spacing x index

where

index = first, first + 1, first + 2, ..., last (if only one contour in the set, first = last)
Signed distance from Mirror/Surface to 0-index contour and Contour spacing are positive in the direction along the positive X-axis

Sometimes you may want to specify a set of contours starting from the information:

Xfirst
Xlast
number of divisions

This is like the first case illustrated above. In such a case, you can choose:

Signed distance from Mirror/Surface to 0-index contour = Xfirst
Contour spacing = (Xlast - Xfirst) / number of divisions
Index of first contour = 0
Index of last contour = number of divisions

In other cases you may know:

Xfirst
Xlast
spacing

This is like the second case above. In this case, you can choose:

Signed distance from Mirror/Surface to 0-index contour = first
Contour spacing = spacing
Index of first contour = 0
Index of last contour = greatest integer less than (Xlast - Xfirst) / spacing

Note that where there is only one contour located at Distance from Mirror/Surface to 0-index contour, you can use Index of first contour = 0, Index of last contour = 0, and Contour spacing = 0 (simple!).

C-spline Entities

Reference for C-spline Curve, C-spline Snake, and C-spline Lofted Surface entities.

Definition

A C-spline Curve is a continuous curve defined by a series of control points. The curve ends at the two end control points (point1 and pointN) and passes through the others in sequence.

Degrees

A degree-1 C-spline Curve is the polyline itself (Fig. 6, top); except for its t-distribution, it is identical to the degree-1 B-spline Curve. For degree-2 and up, the higher the degree, the “stiffer” the curve behaves and the more continuous derivatives it has (Fig. 6, middle and bottom).

More specifically, a degree-2 C-spline Curve is a quadratic spline interpolating the data, with knots midway between the control points. Like a degree-2 B-spline, it is made of parabolic segments that join with continuous slope.

A degree-3 C-spline, like a degree-3 B-spline, is a smooth curve with continuous slope and curvature. With 2 control points, the curve is a straight line. With 3 control points, it is a parabola. With 4 or more control points, it is a parametric cubic spline with knots at the data points, except for the second and second-to-last control points, which aren't knots.

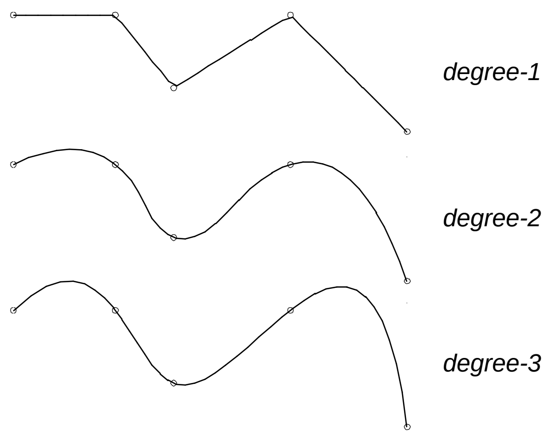


Fig. 6 Three degrees of C-spline Curve.

Technically, a C-spline should have at least one more control point than its degree, e.g. at least 2 for degree-1, 3 for degree-2, etc. However, SurfaceWorks will automatically "demote" a spline to lower degree as necessary to fit the number of control points. For example, a degree-3 spline specified with only 2 points will be treated as degree-1; with 3 points it will be treated as degree-2; only with 4 or more points will it actually be a cubic spline.

Note: Unlike B-splines, C-splines don't work when two or more consecutive control points are identical.

Foil Curves

Reference for Foil Curve entity.

Definition

A Foil Curve is a true NACA airfoil section. Foils can be constructed with 3, 4, or 5 control points — 3 for a half section, 4 for a symmetric full section, or 5 for a cambered or symmetric full section (see Fig. 1). In all cases:

- The first Defining Point is the trailing edges.
- The third Defining Point is the leading edge.
- The first and third Defining Points (trailing and leading edges) form the "chord line".
- The second and fourth Defining Points (top and bottom camber) of a 5-point foil, determine thickness and camber.

- The last control point in a 4- or 5-point foil (second trailing edge) nominally closes the foil. This is usually the same point as the first Defining Point (trailing edge).
- The curve will lie in a plane if the control points do; otherwise it will lie on a parabolic cylinder, with generators perpendicular to the chord line, passing through all the control points

Half-section foils

With 3 control points you get one half of a symmetric foil, with the first point at the trailing edge (t = 0), the second point in the middle, and the third point at the leading edge (t = 1).

In a half-section foil, the second point establishes the half-thickness of the symmetric foil. This is the distance of the second point from the chord line joining the first point and third point. The fore-and-aft position of the second point is immaterial; it is effectively moved parallel to the chord line to the maximum thickness position before being used.

The half-foil curve lies in the plane of its 3 control points.

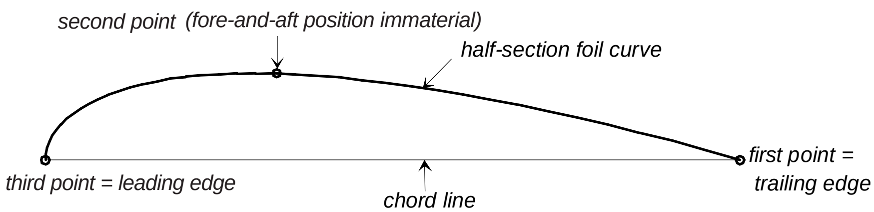


Fig. 7 Foil Curve: leading edge, trailing edge, and chord line.

Full-section foils

With 4 or 5 control points you get a full-section foil — that is, if the first point and the last point on the foil coincide, or are the same point. When the first point and the last point are not the same, you still get a more-or-less foil-shaped curve, but the trailing edge is open.

5 control points. This creates a full section:

- <u>symmetric — when</u> the second and fourth point are symmetrically placed (mirror images) with respect to the chord line
- <u>cambered — when</u> the second and fourth point are NOT symmetrically placed with respect to the chord line

The control points are in this order:

first point — trailing edge (top; t = 0)
second point — top camber
third point — leading edge (t = 0.5)
fourth point — bottom camber
fifth point — trailing edge (bottom; t = 1)

In a 5-point foil, the second point and fourth point act together in determining the thickness and camber. The "upper" surface at the maximum thickness position will be at the same distance from the chord line as the second point. The "lower" surface at the maximum thickness position will be at the same distance from the chord line as the fourth point. The fore-and-aft positions of the second and fourth points are immaterial; these two control points are effectively moved parallel to the chord line to the maximum thickness position before being used. If you place the second and fourth points at the maximum thickness position, the foil curve will pass through them.

4 control points. This creates a symmetric full section from control points in this order:

first point — trailing edge (top; t = 0)
second point — top camber
third point — leading edge (t = 0.5)
fourth point — second trailing edge (bottom; t = 1)

A 4-point foil behaves just like a 5-point foil in which the 4th parent is the mirror image of the 2nd parent across the chord line.

Types

type 1- 5 selects one of five standard NACA foil families (Fig. 2):

type-1	NACA 4-digit series; max camber at 30%
type-2	NACA 63 series with a=0.3 mean line
type-3	NACA 64 series with a=0.4 mean line
type-4	NACA 65 series with a=0.5 mean line
type-5	NACA 0010-34; max camber at 40%

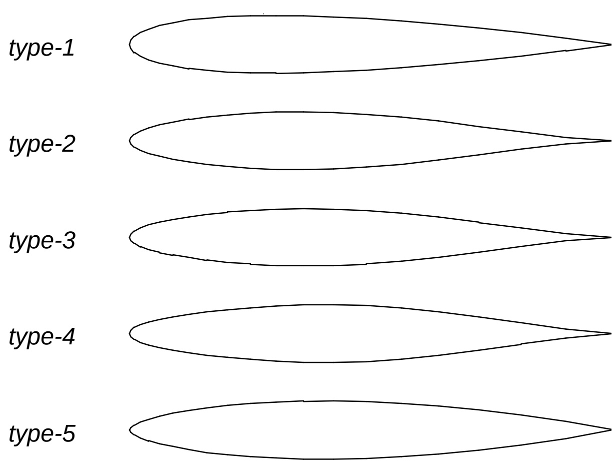


Fig. 8. Foil Curve types 1 - 5.

Type-1 (the 4-digit series) is the workhorse turbulent section, recommended for all applications where surface smoothness and cleanness are operationally limited. With their thickness well forward (maximum at 30% chord) and generous nose radius, the 4-digit foils with thicknesses of 8% to 12% have high maximum lift coefficient and perform well over a wide range of lift coefficients.

Types 2, 3, and 4 (the 63, 64, and 65 series respectively) are progressively more extreme "low-drag" or laminar-flow sections, permitting laminar flow over the forward 30%, 40%, and 50% of chord under favorable conditions of immaculate surface maintenance, low ambient turbulence, and correct lift coefficient range.

Type-5 has the thickness function of the NACA 0010-34 section and the same camber as the type-1 foil.

For more information about the shapes and aerodynamic characteristics of these foil families, see Theory of Wing Sections by I. H. Abbott and A. E. von Doenhoff, Dover Pubs., New York, 1959.

Frame Entities

Reference for Frame entities.

Introduction

In SurfaceWorks, we work primarily in a single global coordinate system with coordinates X,Y,Z. It is a useful simplification to only have to think about one coordinate system, so this usually works well. However, there are certainly times when a secondary coordinate system comes in handy. This particularly arises when designing entities with moving parts: we would like to be able to define a set of entities in a fixed relationship to one another, then later slide and / or rotate the whole assembly as a rigid body to a new position and orientation. Frames are the key to doing this.

Definition

In SurfaceWorks, a frame is an auxiliary coordinate system or “frame of reference” located in the X,Y,Z design space. It is a coordinate system with its own origin, usually different from the global origin. A frame can have any position and orientation with respect to the global coordinate system. We use lower case x,y,z for the frame coordinates. Three ways are provided for defining a frame:

1 You can choose the default frame ‘*’. This is the same as the global coordinate system.

2 You can name any point entity. The frame that results has its origin at that point and is parallel to the global coordinate system.

3 You can create a Frame entity, naming three point entities as parents. Since there are so many ways to locate points, this turns out to be very flexible.

Display

A frame entity is displayed in wireframe views as a small set of three short perpendicular axes. The x-axis is solid, the y-axis is broken into two dashes, and the z-axis is broken into three dashes.

Usage

Using the frame as the Frame parent for points, and these points as the basis for curves and surfaces, you can create assemblies which you can then move intact by just moving the frame they are built in. Another use for a frame is to specify an insert point and orientation for WireFrame entities.

Intersection Entities

Reference for Intersection Ring and Intersection Snake entities.

Introduction

Intersection and projected entities solve various problems involving intersections between objects. The intersection of a snake and a snake on the same surface and the intersection of a snake and a surface (different from the surface the snake belongs to) will generally be a point, which can be made as a ring on one of the snakes. The intersection of two surfaces is a snake which could lie in either surface.

In general, you cannot expect two 3-dimensional curves or two snakes on different surfaces to intersect unless they have been specially constructed to do so. However, two snakes belonging to the same surface can easily intersect at a point. And you can construct two curves to intersect in a variety of ways — for instance, you can put a bead on the first curve and use that bead as one of the control points of a C-spline Curve second curve.

Basic definitions

The basic data for intersection entities are the names of two entities that logically can intersect. We think of the first of these entities as being cut and the other as doing the cutting or being the cutter. The entity resulting from the intersection is a ring or snake on the entity being cut.

The entity being cut

In general, you have two alternative ways of specifying the snake or surface which is being cut:

- to specify a snake, you can name either the snake itself or a ring lying on that snake
- to specify a surface, you can name either the surface itself or a magnet or ring lying on that surface

This flexibility is indicated in the entity specification by slash punctuation, for example: Mirror/Surface or Ring/Snake.

Why name a ring or magnet rather than a snake or surface. To find an intersection, the program must search for it — it sort of casts back and forth (a bit like a hound searching for an elusive scent), refining its search with each step. Whether it gets to the intersection and how accurate its solution is depends on where it starts its search. As you might imagine, it is more likely to be successful if it starts its search close to the intersection.

By default, the program starts searching at the middle of the snake or surface being cut: t = .5 on a snake, or u = .5, v = .5 on a surface. It uses this default when you just name a snake or surface (as opposed to a ring or magnet) when designating the entity being cut. In many cases this will be accurate enough.

In other cases though, especially when there is more than one possible intersection, you may need to give the program a better starting point for its search. You do this by naming a ring or magnet when designating the entity being cut — the point entity performs 3 functions:

- it designates the snake or surface being cut
- it designates which one of multiple intersections you want

- it helps the program find the intersection by giving it a good starting point to begin its search

Place the point entity close to the intersection you want — the closer, the better (within reason — you want the program to calculate the intersection, so don't go to the trouble of calculating it yourself!). For example, let's take the case of an Intersection Ring at the intersection between one surface and a snake on a different surface that crosses the first surface twice. If you were to create a ring and drag it along the snake until it is right next to the intersection you want (placing it by eye should be fine), the ring would both designate the surface being cut and give the program a close starting place to begin its search for the Intersection Ring location.

The cutter

In order to allow cutting flexibility, the actual cutter for most of the intersection entities is designated as a combination of a Mirror/Surface and a Point. This combination defines the "cutting surface" (note that this cutting surface is not a real SurfaceWorks entity; it is only an imaginary construction surface):

Mirror/Surface defines an infinite family of potential cutting surfaces. It can be either a mirror (plane, Line, or point) entity or a surface entity.

Point specifies which one of the infinite family of cutting surfaces is the actual cutting surface. Point specifies the distance from Mirror/Surface to the cutting surface. When Point lies on Mirror/Surface, the distance is zero. Point can be any point object.

The following table shows the effect of the various choices for cutters:

Mirror/Surface	**Cutting Surface (a construction surface; not a real entity)**
plane (Mirror)	the parallel plane which passes through Point — Point specifies the perpendicular distance from the specified plane to the cutting plane
line (Mirror)	the circular cylinder with its axis along this line and which passes through Point — Point specifies the radius of the cylinder
point (Mirror)	the sphere centered at the mirror that passes through Point — Point specifies the radius of the sphere
surface (Surface)	the offset surface which passes through Point — Point specifies the perpendicular distance from the specified surface to the cutting surface When Mirror/Surface is a surface, usually the cutting surface you want is the surface itself, not one of its parallel surfaces. In this case, you can use any magnet on the surface for point; this assures a zero offset.

(A parallel surface is a surface that is spaced a uniform perpendicular distance away from the surface it is based on. An Offset Surface with uniform offset is a parallel surface.)

Intersections

In all cases of calculation of intersections, there is the possibility that the desired intersection is in fact nonexistent. Just because two surfaces might intersect doesn't mean they actually do intersect. Just because a snake can pass through a surface doesn't mean it actually will. Don't worry — non-existent intersections are trapped geometry errors.

At the same time, there is often the possibility that two entities will intersect in two or more places. SurfaceWorks handles this by giving you the option to “point” to the correct intersection with a ring or magnet on the entity being cut.

There is another awkward possibility — that the intersection actually has the wrong dimensionality. For example, take the Intersection Ring, which being a ring, is in the class of point objects. Normally, a surface would be cut by a snake at one or more isolated points. However, it is possible that part or all of the cutting snake actually lies in the surface, so the intersection is really a curve. Similarly, the intersection of two surfaces would usually be a curve; but it could be a surface, if the two happen to coincide over some area. In general, these higher-dimensional, coincidence type intersections will produce non-useful results or geometry errors.

Usage

Intersection entities provide important capabilities in SurfaceWorks. However, we present them with a cautionary note. Because they involve iterative solutions, they are slower to evaluate, less reliable, and somewhat less precise than other entities of the same class. They involve more error possibilities. Often, other entities provide a better — simpler, faster, more direct, or more reliable — way to accomplish a design task. A prime example of this dichotomy is provided by the conventional way of handling surface-surface joins and intersections in 3D CAD:

1. construct the two surfaces so they deliberately intersect
2. solve for the curve of intersection
3. use the intersection curve to trim off the unwanted portions of the original surfaces

You can do this same thing in SurfaceWorks, but it doesn’t mean you should! Very often, surfaces that join accurately and durably can better be made by joining two surfaces to a common edge curve, or by starting a surface from a snake on another surface: you design the intersection first, then make surfaces that join correctly onto it.

Lofted Surfaces

Reference for Arc Lofted Surface, B-spline Lofted Surface, and C-spline Lofted Surface entities.

Definition

SurfaceWorks recognizes three kinds of lofted surfaces: Arc Lofted Surface, B-spline Lofted Surface, and C-spline Lofted Surface.

A lofted surface is shaped by two or more control curves in much the same way as a curve is shaped by two or more control points. The process (which SurfaceWorks carries out automatically and invisibly) is:

1. Decide the orientation of the control curves with reference to the first control curve that has positive length.
2. On each of the control curves, locate points at the same parameter value t (or 1 - t, if the curve is used with reversed orientation).
3. Use the resulting series of points as the control points for a lofting curve.

The curve used in step 3 is a circular arc, B-spline curve or C-spline curve depending on your choice of Arc Lofted Surface, B-spline Lofted Surface or C-spline Lofted Surface. An Arc Lofted Surface's behavior is decided by the type of Surface chosen (there are 6 available). A B-spline Lofted Surface interpolates its start (curve1) and end (curveN) control curves, but in general not the others. A C-spline Lofted Surface passes through all its control curves.

One or more control curves of a lofted surface can be a point entity (point, bead, magnet, or ring).

NUB and NURBS Entities

Reference for NURBS Curve, NURBS Snake and NURBS Surface entities and NUB-spline Fitted Curve, NUB-spline Fitted Snake and NUB-spline Fitted Surface entities.

Definition

The NURBS entity is a Non-Uniform Rational B-Spline entity. The control points guide or shape the entity in much the same way as the control points shape a B-spline entity. In general, the entity does not interpolate any of its control points except the first and last.

The NUB entity is a Non-Uniform B-spline entity. It operates in the same fashion as a NURBS entity with the exception that it does not have weights (the Rational part) to control its shape.

Weights

The weights (wt1, etc.) make the “Rational” part of a NURBS Curve. If weights are all the same and knot spacing (see below) is uniform, you just get a uniform B-spline curve (identical to B-spline Curve). Increasing wt on one control point draws the curve toward that control point. You can think of the curve as being connected to the interior control points by springs; wt is analogous to the stiffness of the individual spring (Fig. 9).

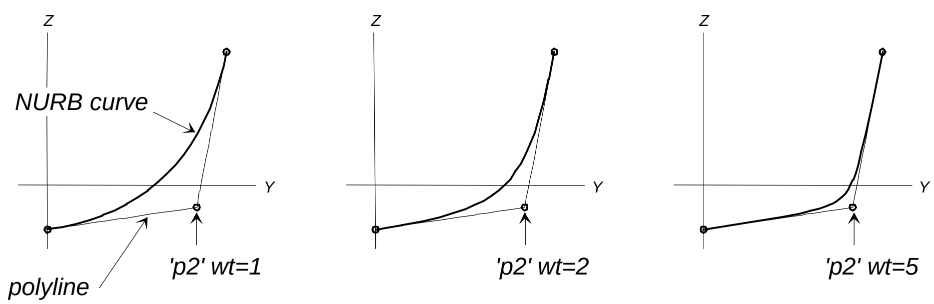


Fig. 9. A NURB Curve with ‘p2’ weight = 1, 2, and 5.

When you change the parents list (control points or curves) for a NURBS entity, its weights are reset to 1.

Note: A control point with a weight of 0 (zero) has no effect on the shape of the curve or surface, but it is possible to get into trouble using weights of 0. For example, in a type-1 NURBS Curve, a weight of 0 technically causes a division by zero and so causes a geometry error. In like manner, in a NURBS Surface with type-1 for both the u and v directions, a single weight of 0 causes the same problem. In a type-2 NURBS Curve, two consecutive 0 weights cause the same problem, etc.

Knot spacing

Knot spacing, as specified in a Knot List, is used in both NURBS and NUB-spline Fitted entities.

Non-decreasing numbers

The list of values must be composed of non-decreasing numbers.

First and last value

The list should begin with 0 and end with 1.

Number of values

The number of values required varies according to the number of control points and the Degree of entity involved. The Knot List should have: (number of control points) - (type) + 1 members. For example, if the number of control points = 6 and Degree = 3, you'd need 6 - 3 + 1 = 4 members; { 0 .25 .75 1 } would be a possible set of values.

Spacing type

The spacing can be either uniform or non-uniform. For uniform spacing, just use the default '*'. For non-uniform spacing, any set of values that follows the above rules will do. Drawing knots together allows the curve (or surface) to have high curvature in the vicinity of the multiple knots. At most Degree + 1 sequential knots can have identical values.

Types

Technically, a NURBS entity should have at least one more control point (or curve) than its Degree, e.g. at least 2 for Degree-1, 3 for Degree-2, etc. However, SurfaceWorks will automatically "demote" a spline to lower Degree as necessary to fit the number of control points. For example, a Degree-3 spline specified with only 2 points will be treated as type-1; with 3 points it will be treated as type-2; only with 4 or more points will it actually be a cubic spline.

Parametric Solids

All of the solids you can form with SurfaceWorks are defined with reference to a trio of parameters — u, v, and w — each of which, has a nominal range of 0 to 1. The rule that defines the solid in terms of its basis surfaces and/or curves is written in terms of these three parameters. You can think of every point of a solid as being labeled with a particular trio of values for u, v, and w. Generation of the solid is a continuous mapping of points in the unit cube to points in 3D space. We refer to the unit cube as the "u-v-w parameter space" of the solid. It is a 3D space, each of whose points correspond to a point in the solid.

At any point in a parametric solid there is a direction (called the "u-direction") in which u increases while v and w are constant. Similarly there is a v-direction in which v increases while u and w are constant and a w-direction in which w increases while u and v are constant. The only reason you usually need to be conscious of which direction is which is when deciding how finely the solid should be subdivided in the three directions for tabulation and display.

A parametric solid nominally has six faces and eight corners. The eight corners have (u,v,w) values of (0., 0., 0.), (0., 1., 0.), (0., 0., 1.), (0., 1., 1.), (1., 0., 0.), (1., 1., 0.), (1., 0., 1.), and (1., 1., 1.). Some of the faces can be degenerate surfaces, i.e. curves or even points.

Just as curves and surfaces exist beyond their nominal endpoints, most SurfaceWorkssolids extend smoothly outside their nominal parameter range of 0 to 1. You might visualize this by imagining extruding each of the faces of the solid outward from the solid.

The distribution of the parameter t along the curves used to build surfaces supporting a solid affect the shape of the solid. In most cases, the “default” labeling works just fine. For a discussion about changing the labeling, see "Editing Models - Relabeling Curves and Snakes".

For some applications and examples of using solids, including how to export 3D meshes, see "Using Solids" on page 121.

Note: SurfaceWorks users, please understand that support of parametric solids does NOT turn SurfaceWorks into a “solid modeler” comparable to the design programs in that category such as Pro-Engineer, SolidWorks, and Solid Edge. These solid modeling applications are based on B-rep (“boundary representation”) solids, a very versatile way to represent solid objects of arbitrary complexity. Parametric solids are a more restricted solid representation in that they must have the basic 6-face topology of a cube (just as parametric surfaces always have the 4-sided topology of a square). On the other hand, the solid shapes you can make with SurfaceWorks parametric solids are comparatively diverse because SurfaceWorks surface modeling capability is much richer than that of most solid modelers.

Additional Solid Attributes

divisions: u, v, and w divisions and subdivisions

All solids have u-divisions, u-subdivisions, v-divisions, v-subdivisions, w-divisions, w-subdivisions attributes. The number of u (or v or w) divisions x subdivisions = the number of line segments the program uses to represent the surface in the u (or v or w) direction for display and internal tabulation. u-divisions, u-subdivisions, v-divisions, v-subdivisions, w-divisions, and w-subdivisions are all integers (1-255). The greater the total number of u, v, and / or w divisions, the more closely the solid screen display will approximate the solid itself (which is composed of an infinite number of points).

In addition, the number of u-divisions (or v- or w-divisions) establishes the spacing of the u-constant (or v-constant or w-constant) lines drawn in a wireframe display.

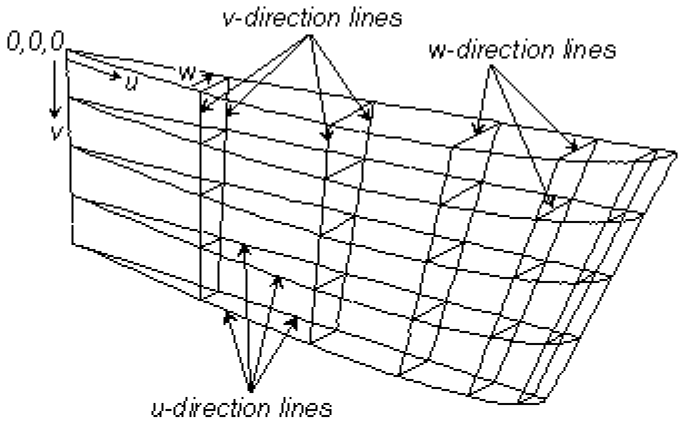


Fig. 10. u, v, and w-direction lines on a solid.

For instance, if a solid is defined with 6x4 4x2 2x2 divisions, SurfaceWorks will tabulate the solid internally with 24 segments in the u-direction, 8 segments in the v-direction, and 4 segments in the w-direction (Fig. 4).

Note: The visibility options for solids do not "parallel" those for surfaces. Solid display options show the u-direction lines (not the u= constant lines), etc. These display lines run in the same direction as the orientation marks for u, v, and w

Orientation

Like surfaces, solids have an orientation attribute which can be either 0 (normal) or 1 (reversed). Solid orientation is different from surface orientation — currently its only effect is to reverse the sign of volume and weight.

Using Solids

Some applications

- Accurate weight analysis and CG for sailboat ballast (see RuledSolid examples 1 and 2)
- Tank volumes (for an example, look at the model Tanks.ms2
- Exterior volume meshes for CFD analysis (for an example, look at the model Exterior-Grid.ms2.
- Interior volume meshes for finite element analysis (for an example, look at the model Interior-Grid.ms2.

Discussion of tank and grid examples

Tanks.ms2

This is a model for an integral tank that can be dropped into almost any hull with a single-surface bottom. It starts with 'p0', which sets the positions of the planes forming the flat forward, top, and inboard surfaces, and 'p1' which controls the length of the tank. The line 'l0' between these points is projected horizontally and vertically onto 'hull'. Each of the resulting ProjSnakes gets a ring at each end ('r00', 'r01', 'r10', and 'r11'). Lines between the rings are projected onto 'hull', then a SubSurf 's4' is made between the four ProjSnakes.

To make the RuledSolid 'fwd_tank', we need two surfaces. One is the SubSurf 's4' and the other is a degenerate RuledSurf between the line 'l0' and itself.

Tools / Mass Properties gives the tank volume = 7.68 cubic ft.

Tools / Weight Schedule gives the weight (and c.g.) of its contents = 958 lbs including the port-side image.

Exterior-Grid.ms2

'mesh1' is a BLoftSolid forming a body-fitted 3D mesh for CFD analysis of flow past a streamlined body. The 'body' (bright green) is a BLoftSurf made from elliptic cross-sections. The BLoftSolid has three control surfaces: 'body', 'middle' (bright blue; an OffsetSurf), and 'outer' (dark red; a cylindrical outer-boundary surface). Using an OffsetSurf for 'middle' ensures that the mesh is orthogonal to 'body'. The mesh is also relabeled in the w direction, creating thin cells at the 'body' surface for viscous flow analysis.

'mesh1' has 8x4x8x4x8x2 = 16,384 cells in each quadrant.

Interior-Grid.ms2

This model is a volume grid for finite-element analysis of a pulley wheel. (Turn on symmetry or View / Render to see all the images.) 'grid' (gray; on Layer 2) is a RuledSolid between the inner and outer RevSurfs (bright green; on Layer 1).

If you turn layers 1 and 2 off and take the <y> view, you can make many parametric variations by dragging points 'p0', 'p1', 'e0', and 'p3'. The analysis grid automatically adapts.

Procedural Entities

Reference for Procedural Surface

Definition

A procedural entity is defined by creating one typical point (or curve) of the procedural entity and then having SurfaceWorks repeat the point (or curve) creation procedure for all positions of the parent beads / rings / magnets along the supporting bead / rings / magnet's host curve or surface. As essentially user-defined entities, the procedural entities vastly broaden the flexibility of SurfaceWorks' relational modeling.

The "moving entity" (the typical point or curve) that sweeps out a procedural entity is constructed in some fashion from magnets or beads / rings, and it depends on them directly or indirectly. The magnets / beads / rings must be Magnets, Beads and Rings that are not relative to another entity. These parents can be all of one kind (e.g. all Beads or all Rings) or they can be mixed (e.g. two Beads and a ring; or a Bead, a Ring, and two magnets).

If the bead / ring parents don't all have the same t, the program preserves the t offsets between them. Similarly for magnet supports with differing u,v. The first bead / ring or magnet is the "master" one, which is varied from 0. to 1.

To generate the procedural entity, the program varies the bead / ring (t) or magnet (u,v) parameters from 0. to 1., and the procedural entity is the resulting path / surface of the moving point or curve.

Note: Before starting the construction of any procedural object, SurfaceWorks records the locations of the "moving object" and all parent beads, rings, and / or magnets. After construction, it restores all these parents and their children to their specified locations and configurations.

Projected and Mirrored Entities

Reference for Projected Point, Projected Curve, Mirrored Point, Mirrored Curve, and Mirrored Surface entities.

Note: Projected Magnets and Projected Snakes work somewhat differently from the other projected entities. We've put all of their specification information in the entity descriptions.

Definition

Projected Point and Projected Curve entities are formed by projecting a specified basis entity onto a Mirror. Each point of the basis entity is projected normally (i.e., perpendicularly) onto Mirror. The projected entity therefore lies entirely in Mirror.

Mirrored entities are created by reflecting a given basis entity through a Mirror. Each point of the basis entity is projected normally (i.e., perpendicularly) through Mirror, to a point equidistant on the other side. The mirrored entity therefore lies on the opposite side of Mirror from the basis entity.

Comparison

The data categories defining mirrored points, curves, and surfaces are identical to those for projected points, curves, and surfaces: basis entity and Mirror. The difference lies in how the data is used. In each case, the new entity consists of points which have been projected along temporary lines perpendicular to Mirror. A point on a projected point, curve, or surface would be located at the foot of its projection line, where it meets the Mirror. For a mirrored entity, each projection line is extended an equal distance on the far side of Mirror, and the mirrored entity is located there.

Mirror

In either case, Mirror can be a plane, Line, or point entity:

When Mirror is a plane, each projection line is perpendicular to the plane.
When Mirror is a Line, each projection line is perpendicular to the Line.
When Mirror is a point, each projection line runs through that point.

Triangle Mesh Entity Types

SurfaceWorks has partial support for a new class of entity types called Triangle Meshes. A Triangle Mesh is a form of surface, fundamentally consisting of an assembly of triangles that join edge-to-edge. A Triangle Mesh can also be subjected to an optional smoothing / refining operation known as subdivision, which produces successively finer meshes. Points and curves can be embedded in a Triangle Mesh, analogous to magnets, snakes and rings on a surface.

Triangle meshes are much less constrained topologically than parametric surface patches. They can have holes, and any number of sides, or even no sides -- seamless closed bodies are easy to make.

Triangle meshes are included in Surface Works 5.0 primarily for the possibilities they offer in flattening with the add-on Flattener product. An STL or RAW file can be imported, resulting in a Light Triangle Mesh. So far, they have no other import and export options.

Triangle Mesh basics

The basic constituents of a Triangle Mesh are a set of points, called nodes, and a set of triangles, each made from 3 distinct nodes. The nodes are numbered with an index, 1 to nnodes. The triangles are numbered with another index, 1 to ntris.

Each triangle has 3 edges -- lines connecting 2 nodes. Each edge can be shared with at most one other triangle. An edge that is not shared with another triangle is a boundary edge, and is part of the boundary of the Triangle Mesh.

The triangles must be sufficiently connected by shared edges so you can reach any triangle in the set from any other by stepping from one triangle to the next, always crossing shared edges. For example, two triangles sharing just a single node do not make a valid Triangle Mesh, nor do two triangles that don't share any nodes.

Unlike a surface, which has an invisible extension outside the 0-to-1 parameter range, a Triangle Mesh has no extension; when you come to the boundary, you're really at the boundary.

Recursive subdivision

Subdivision is an operation applied to a Triangle Mesh, that results in another Triangle Mesh with 4 times the number of triangles. The nodes for the subdivided mesh are derived from the original nodes by an averaging or smoothing operation

In each stage of subdivision, a new node is generated near the middle of each existing edge, and each original triangle is replaced by 4 triangles.

Recursive subdivision is the repeated application of the subdivision operation. Any Triangle Mesh entity has a degree property that controls its degree of recursive subdivision. degree is a non-negative whole number. The default value is 0, which means "no subdivision". The unsubdivided mesh is often referred to as the "coarse mesh".

Locations on a Triangle Mesh

A triangle mesh does not have an overall coordinate system like the u,v-parameters of a parametric surface. However, within each triangle it has a local coordinate system that can be used to locate points on the surface. Locations in a triangle are specified by so-called barycentric coordinates u, v, w which act as weights applied to the 3 corners of the triangle. In the triangle interior each of u, v and w is in the range of 0 to 1, and they satisfy the constraint: $u + v + w = 1$, or (equivalently) $w = 1 - u - v$.

Thus the combination {triangle index, u, v} is sufficient to uniquely identify a point located on a Triangle Mesh, because w can be expressed as a function of u and v.

Figure: Barycentric coordinates in a single triangle

Common data

Orientation. Any Triangle Mesh entity has a normal orientation property with one of the two values: Normal and Reversed. This is very similar to the orientation property of a surface; it just defines which direction will be considered the positive normal.

Weight/unit area. A Triangle Mesh has a weight / unit area property that is completely analogous to weight / unit area for a surface. It is multiplied by the Triangle Mesh area for inclusion in the Weight Schedule.

Visibility options. The edges of component triangles can be made visible or hidden.

Kind. A Triangle Mesh has a kind, which is not yet used for anything. The only valid value for kind is 0.

Degree. The degree of subdivision, a nonnegative whole number.

Snakes on Triangle Meshes

There is a class of Triangle Mesh Snakes representing curves embedded in a Triangle Mesh. These are highly analogous to snakes on surfaces. There is a variety of possible ways to specify them, represented by 5 different entity types.

NOTE: Triangle Mesh snakes are curves, but for technical reasons in Surface Works 5.0 they can't serve as curve parents for other entities (for example, as master curves for a surface).

Surfaces as Triangle Meshes

Since the 5.0 release, SurfaceWorks has supported a family of Triangle Mesh entity types, alongside the Surface class of entities. Triangle Meshes are an alternative surface representation with significant advantages and disadvantages compared with parametric surfaces. The biggest disadvantage is that a Triangle Mesh is always a faceted surface, and therefore not as smooth as most parametric surfaces. The most important advantage of Triangle Meshes relative to parametric surfaces is freedom from the topological constraints that arise from construction in relation to a [0 to 1] x [0 to 1] u,v parameter space. While a parametric surface is basically a 4-sided object, a Triangle Mesh can have any number of sides, or none at all (a seamless closed body).

Triangle Meshes have been implemented so far as a largely separate world from parametric surfaces. They have their own family of snakes, magnets and rings. The SurfTriMesh entity type has been a partial bridge between the two worlds, allowing a Triangle Mesh to be derived from a surface, Trimmed Surface or Composite Surface entity.

This release significantly expands the Triangle Mesh family with 5 new entity types. A related visible change is that we have substituted "TriMesh" for "Triangle Mesh" on menus, prompts and property labels throughout the program.

A less visible change is that, with this release, much of the gap between the "two worlds" has been paved over. This began with the realization that in almost every case where SurfTriMesh was used, we were taking the default properties nu = 0, nv =

0, style = 0. The SurfTriMesh looked like an extra, unnecessary step. Why not just allow any surface, Trimmed Surface or Composite Surface to be used as a triangle mesh? This turned out to be easy to do, and invited the further rules:

Any magnet or ring on a surface can be used as a triangle mesh magnet

Any snake on a surface can be used as a triangle mesh snake

The triangle mesh you get from an untrimmed parametric surface is divided along the surface's uniform division and subdivision lines, then each resulting rectangle (in u,v parameter space) is divided along one diagonal, to make two triangles. Thus the total number of triangles for one surface is (assuming no degenerate edges):

u-divs x u-subdivs x div-mult x v-divs x v-subdivs x div-mult x 2

A simple way to visualize the triangles you'll be using is to make a CopyTriMesh from the surface, with * and * as the two frames.

A CopyTriMesh also becomes a handy answer to the problem of mirroring or copying a Trimmed Surface or Composite Surface, as long as a triangle mesh is sufficiently smooth and accurate for your purposes.

Several of the new entity types involve copying information from one triangle mesh to another: TriMesh Copy Magnet, TriMesh Copy Snake, and Copy Contours. These require a high degree of correspondence between the two triangle meshes, termed "congruence". To rate as congruent, two triangle meshes need to have not only the same number of nodes and triangles, but also identical connectivity and indexing -- so for example, triangle 37 is the "same" triangle in each mesh, and has its local u and v directions corresponding on the two meshes.

Example files

SurfaceAsTrimesh1.ms2

Example of Surface entity serving as a TriMesh. Revolution Surface s1 is a portion of a cone; it supports Expanded Trimesh tm1 (its plane expansion).

TrimSurfAsTriMesh1.ms2

Example of Trimmed Surface 'trim' serving as TriMesh. CopyTriMesh 'tm2' is made from 'trim'; 'trim' and 'tm2' are combined in PolyTriMesh 'tm3'; 'tm3' is copied 6 times into different frames to make the snowflake design.

CompSurfAsTriMesh1.ms2

Example of Composite Surface serving as a TriMesh. CompSurf L0 is made from 3 square B-spline surfaces. CompSurf L0 supports 8 TriMesh Magnets and a TriMesh BSnake. CompSurf L0 also supports an OffsetTriMesh, which has a copy of the snake.

Using SurfaceWorks Effectively

Guidelines for Designing in SurfaceWorks

We hope this set of guidelines will help you design faster and better in SurfaceWorks. We envision it as a summary-type reference you might read several (or numerous?) times over the course of learning to use SurfaceWorks, absorbing more of it each time. (Please don't ask yourself to absorb it all at one sitting!!) As with any tool, we are convinced that practicing good SurfaceWorks habits from the start will shorten your learning curve, as well as increase your creativity and productivity later on.

Note, though, that there are many different ways to do things in SurfaceWorks. These guidelines represent the collective knowledge and experience of the people at AeroHydro, gained from creating our own models and from helping other people with their models (frequently much more complex than our own). While you can gain much by following such guidelines, they are not the law, merely suggestions. It is always beneficial to try new things, for even if they don't work, you will have learned something new.

1 & 2. Do and Use

Do the tutorials!

Use, use, use — there is no substitute for experience using SurfaceWorks. With experience, all these hints will become self-evident and second-nature.

3. To Get Started

To get started, make a quick, initial sketch on paper of what you want your model to look like and make a few speculations about some major relationships you would establish, then get going — make what you know has to be there, that is, start with the major curves. At this beginning stage, keep your rough plan hovering but don't try to fill in all the details before you get started (or you'll never get started)!

Example

I want to design a free-standing bathtub, to be constructed of fiberglass. This is to be the first in a line of fiberglass tubs that imitate and improve upon the old cast-iron tubs. I make my rough sketch, noting leading dimensions and important curves. I know that the shape of the feet will be independent of the tub (they will attach to the body of the tub), so I can start work on the tub itself and not worry about the feet. Since the tub will be symmetrical across its longitudinal centerplane, I can design one half of it and mirror the other half.

I will keep in mind ways to easily change the leading dimensions and basic shapes by building parametric handles into my model. In this way, I can easily modify the model to make a different size unit or next year's tub line.

The important curves on the tub are its upper lip, the two end slants, and some tub cross-sections, so I'll start by designing them. For the lip, a longitudinal curve, I'll use a C-spline Curve. Then I'll use this curve to define the upper ends of the cross-section curves. For the transverse cross-sections, I'll use B-spline Curves. Then I'll use these curves to define a B-spline Lofted Surface for the first half of the tub and a Mirrored Surface for the other half.

Along the bottom of the tub, I can ensure tangency across the centerplane of the B-spline cross-section curves by putting the bottom two control points of each curve on a line normal to the centerplane at the point where the curve meets the centerplane. That's enough to get started.

4. Balance

There is always a balance to strike between the design modes of "stopping and thinking about things" and "plowing ahead." The big picture can be overwhelming if you try to capture all its details all at once — on the other hand, if you pay no heed to the overall design, you can easily work yourself into a corner or a tangle. You'll probably find you'll use both modes, more than once, at different times during your design cycle.

5. Simplicity

Simple is almost always better — try to use as few entities as possible. Using lots of control points rarely (if ever) gives you the exact curve shape you seek; in fact it can cause unexpected humps and / or hollows.

Example

Situation: I want the shapes of two B-spline Curves to be similar. I find myself making the first B-spline Curve with 14 control points; then making relative Points off all those control points, and using those relative Points as control points for my second B-spline Curve.

Is there another way?: Yes, I could try a B-spline Curve and a Relative Curve off that B-spline Curve — I'll just need two relative Points, not 14. A Copy Curve is another alternative.

6. Named Layers and Entity Naming Conventions

Especially if your model has a large number of entities, use named layers and clear entity naming conventions to make your model more accessible to yourself and others. A little time spent at this task can save you and others hours of time later on. If everyone in your office uses the same set of conventions, everyone will know how to understand models that are passed around and worked on by several people.

Examples

Use names that help you associate points and curves with their child surface; e.g. if you are making a sunroof glass surface you will call 'sunroof_glass', you might name the parent points 'sunroof_g@p1', 'sunroof_g@p2', etc.

Put different parts of a model (e.g. main body, fenders, doors, sunroof) on different layers and keep to the same convention for all similar models. For instance, if you design car exteriors, you might always put the main body on a layer named Body, chrome on a layer named Chrome, etc.

7. Shortcuts

Capitalize on filters, layers, toolbar and keyboard shortcuts, entity names, and quick point and quick spline modes to help you streamline selecting, creating, editing, and manipulating entities. Find a combination that feels comfortable to you.

8. Joining Surfaces Precisely

If you need two surfaces to join precisely, create one surface, construct a snake on it where you want the second surface to join it accurately, then build the second surface off the snake.

Example

Fin-fuselage join — use a Projected Snake on the fuselage. This will durably attach the fin to the fuselage, but will leave the shape of the fin unaffected should you change the shape of the fuselage.

9. Lofted Surfaces

For lofted surfaces, design all the edges first.

Example

For a C-spline Lofted Surface panel, design the first and last control curves plus the curves that would outline the other two edges. Then plan where the intermediate control curves can go, to make the surface have those edges.

The need to begin with the first and last control curves is probably quite apparent. Less apparent is the fact that designing the other two edges will help immensely when you begin to locate the internal control curves.

10. Surfaces with Complex Boundaries

If you are designing a surface with a complex boundary, first make a fair basis surface (using whichever kind of surface gives you the overall fair shape), then trim out the surface you want with a SubSurface (parametric) or a Trimmed Surface (non-parametric).

11. Establishing Tangency

When you need durable, tangent joins between curves or surfaces, keep in mind:

- The tangency properties of B-spline Curves and B-spline Lofted Surfaces
- The G1 and G2 continuity available with either Blend Surfaces or Tangent Boundary Surfaces. See "Creating Entities - How to... - Degrees of Continuity Between Surfaces and Between Curves".
- Using Mirrored entities

For some examples, see "Understanding SurfaceWorks Entities - Collective Entity Information - B-spline Curves - Degrees" and the tutorials.

12. Making Highly-Contoured Surfaces

If you want to make a highly contoured surface, take advantage of the Tangent Boundary Surface. This surface has a user-defined number of interior control points which can be used to push and pull on a surface. It also supports G1 and G2 continuity with adjacent surfaces.

13. Divisions

Divisions — the issues here are speed (of display, evaluation, and conversion) and accuracy:

Number of divisions: For speed, choose a smaller number of divisions, for accuracy, choose a larger number. It is best to estimate low. You want enough to be accurate without reducing speed unacceptably.

Coordinating divisions: For accuracy and for speed of SurfaceWorks-to-IGES conversion, coordinate divisions. The easiest way to coordinate divisions between surfaces and their parent curves is to match their divisions x subdivisions (see "Understanding SurfaceWorks Entities - Defining Entities - Additional Surface Properties - Coordinating Divisions").

Tip: You can set the same divisions for a whole group of curves (or surfaces) by using the multiple edit feature. When the Selection Set is a group of curves (or surfaces), **Edit>Definition** does a multiple entity edit that includes the divisions attribute.

Tip: Remember that the Divisions Multiplier (> 1) will effectively increase subdivisions on all curve, snake and surface entities within the model.

14. Trying to Match Something Very Exactly

If you're zooming in to drag a point and trying to match something very exactly (e.g. to put the point on top of another point), STOP for a moment and consider: how could I do this in a durable and exact way?

Example 1

Situation: I have designed one curve. Now I am designing a second one that I want to end at the same place as the first one. I'm trying to put my new endpoint exactly on top of the first curve's endpoint.

Is there a better way?: Yes, use the same point as the endpoint for both curves. Don't make a new endpoint for the second curve.

Example 2

Situation: I have a lens-shaped surface bulging up from a horizontal elliptical base. I want a point on it at X=100mm, Y=30mm. So I'm trying to drag a Magnet on the surface to exactly X=100, Y=30.

Is there a better way?: Use a Projected Magnet. For the basis point, create a Point at X=100, Y=30, Z=whatever (say 0); and for the mirror, use the predefined '*Z=0' plane (select it from the Available Entities pane). An XYZ Magnet is a new entity, which was added in version 5.0, will certainly fulfill the requirements.

Example 3

("bead" and "curve" are used in the following explanation, although this can also be applied for "ring" and "snake")

Situation: I've created a spline that pierces an arc. I bring them into SurfaceWorks and want to place a bead exactly on the point where the end of the spline pierces the arc, so I can subdivide the arc and use that section as a bounding curve for a Tangent Boundary Surface.

Here's a good way: Select the arc (the curve I want to place the bead on) and the spline. Make an Intersection Bead. The bead will be placed on the first curve in the selection set, in this case the arc. The arc now has a parent bead for inserting a SubCurve.

15. Multi-level Dependencies

Complex, multi-level dependencies are not necessarily bad — sometimes you really need them; but they can certainly complicate editing and maintaining a model. If you find yourself deep into a dependency tree, consider — can I do this task differently?

This is a good point to step back and look at the big picture.

16. Tools for Those Inevitable Changes

Expect to make changes. While it's important to look at the whole picture and plan ahead, none of us ever gets it all right the first time around (unless the project is very simple, and still often not even then)!

Examples

Situation: I have a lofted surface for which all the control curves are based on points that have default Parents. I discover I want to move one control curve forward and another one aft, to better control the shape (and I imagine the scenario of not getting them in exactly the right position the first time). I realize that if each control curve were defined by one Point with default Parents and the other Points using this point as their Point Parent, all I'd need to do to move a control curve would be to drag the one Point.

Solution: For each curve, **Select** all but one control point and in the Multiple Entity Property Manager, make the Point Parent the Point you do not change. Be sure to select "Preserve Absolute Location" to keep the points initially in their current location.

Situation: I want to change a curve parent for a surface from a C-spline Curve to a B-spline Curve, so I can use the tangency property of B-splines to ensure tangency across the centerplane. The C-spline has lots of children and I don't want to lose those relationships, but transferring the children one by one would be a big job.

Solution: You can make a B-spline Curve using the same control points as the C-spline Curve, and use **Tools>Adopt Children** to move all the C-spline's children and relationships to the B-spline. Be sure to check "Delete old parent" if you will have no further use for the C-spline. To finish, adjust the control points (and add or remove control points, as needed) so the B-spline has the shape you want.

17. Clean up Your Model

Clean up your model: When you are getting near completion or when you replace one or more entities, get rid of the old unused ones — if they have no further function, they'll just get in the way and confuse you (or others). Here are steps you can follow:

Turn all layers on (**Tools>Layers>Set All**).
Display only the surfaces you want to keep.
Select>Visible to set the visible only filter.
Select>All.
Select>Parents>All Generations.
Select>Invert Selection Set.
Review the contents of the Selection Set pane (these are the proposed extraneous entities; edit the list if need be, eliminating entities you want to keep from the Selection Set).
Edit>Delete (if the Selection Set includes any entities that have dependents, SurfaceWorks will tell you and it will not perform the delete; to finish, you'll need to remove these entities from the Selection Set, recheck it, then delete again).

If you have other tips to share from your experience, please send them to us.

Reference Guide to Functions

Introduction

This chapter describes the SurfaceWorks menu and toolbar button functions. The functions are listed in alphabetical order, by function (e.g., About SurfaceWorks ... Zoom to Selection). Each function description includes:

- where to find it:
  toolbar icon (if available)
  menu entry
  keyboard shortcut (if available)
- when it's available and how it works
- step-by-step instructions

About SurfaceWorks

Displays version and build number and copyright information for SurfaceWorks / MultiSurf . Choose **Help>About** SurfaceWorks.

Adopt Children

Tools>Adopt Children or **Shift+A**

[Only available when two interchangeable entities are selected. Not available if there is any non-adoptable entity present.]

SurfaceWorks attempts to transfer all the children of the first entity (old parent) and make them children of the second entity (new parent).

Usually used when you have created a replacement entity for an existing entity that has many children. This saves the effort of changing the parent of each child one at a time.

In order for a child entity to be adopted by the new parent, new parent must qualify to serve as a parent for that entity (e.g. a curve can directly adopt beads and surfaces but not magnets or rings) — usually this means that new parent and old parent must be in the same entity class (e.g. both surfaces). If there are any entities in the old parent that cannot be supported by the new parent, the process will not work.

SurfaceWorks will transfer the children of old parent to new parent, but it won't be able to transfer a parent if that would cause cyclical dependency. If old parent is left with no children, it can be deleted.

To transfer children from one entity to another:

1. **Select the "old parent"**, the entity from which you want to move child entities.

2. Then **Ctrl+click the "new parent"**, the entity which is to adopt the children.

3. Choose **Tools>Adopt Children** or press **Shift+A**. (If the pair of entities you have selected do not qualify for child swapping, the Adopt Children option will be grayed out.)

4. **Check** that the entities you selected are playing their intended roles. If need be, click the Reverse button to switch their roles. (If the pair of entities do not qualify for role reversal, the Reverse button will be grayed out.)

5. **If you want to delete the old parent**, check the box.

6. Click **OK**.

Delete old parent. Check the box if you want the old parent entity deleted from the model after its children have been adopted. This option will be grayed out when not all of the children can be adopted by the new-parent entity.

All

See " "Select All" on page 210".

See "To hide All:" on page 178

See "To show All:" on page 217

All Generations

See "Select Children" on page 210"

See "Select Parents" on page 212 .

Angle

Tools>Measure>Angle

[Only available when three point entities are selected.]

Measures the angle at the middle point in the Selection Set (point2) between the directions from the middle point to the first point in the Selection Set (point1) and to the third point in the Selection Set (point3). (Mnemonic: this is similar to geometry's notation "angle ABC" for "the angle at B between the directions to A and C".)

Angle measures are in degrees. The order of the points in the Selection Set is important.

To measure angle:

1. Select the point **along one side** of the angle.

2. Add to the Selection Set the point **at the apex** of the angle.

3 Add to the Selection Set the point **along the other side** of the angle.

4 Click  or choose **Tools>Measure>Angle**.

Arrange Icons

Window>Arrange Icons

Arranges the icons for minimized SurfaceWorks windows in a row along the bottom of the SurfaceWorks workspace.

To line up the minimized window icons:

Choose **Window>Arrange Icons**.

Auto Orientation

Tools>Options>General

Provides the option to automatically display orientation marks while an entity is selected. The key words here are “automatically” and “while selected” — they distinguish Auto Orientation from the non-automated Orientation that you can pin to an entity.

Auto Orientation is an application setting.

Note: If you want to display orientation marks for an entity of your designation regardless of its selection status, use the non-automated View>Entity Orientation. (See "Entity Orientation" on page 164).

To automatically display parameter orientation information (or to turn the auto display off):

1 Go to **Tools>Options>General**.

2 **Click the radio button** option of your choice.

None. Auto display is turned off. No orientation marks are automatically displayed when an entity is selected.

Single selected entity. When the selection is a single entity, its orientation marks are displayed. This is the initial program default.

All selected entities. Orientation marks are displayed for all selected entities, no matter how many entities are selected.

The Entity Orientation Display

Parameter orientation is displayed in two ways, depending on what kind of entity is selected:

Curves, snakes: A circle is drawn around the t=0 end of the curve; and an arrowhead is drawn halfway along the curve, pointing in the direction of increasing t.

Surfaces: A circle is drawn around the u=0, v=0 corner of the surface; a "u" (for u-direction) and an arrowhead are displayed halfway along the v=0 edge, and a v (for v-direction) and an arrowhead are displayed halfway along the u=0 edge. An arrow, emanating from the center of the surface, points in the direction of the positive normal to the surface. On Trimmed Surfaces only the positive normal is displayed.

The orientation marks are displayed in black.
Auto orientation marks (see "Tools Options" on page 224).
Non-automated orientation marks (see "Entity Orientation" on page 164).

Axes

View>Axes or **A**

Toggles display of world coordinate axes on and off. When the axes are on, the menu option is preceded by a checkmark.

Axes are reset to on each time a model is opened.

To toggle display of the world coordinate axes:

Choose **View>Axes** or press **A**.

Both

See "Select Both" on page 210

B-spline Curve Fit

Tools>Special>B-spline Curve Fit

By invoking this function you have the ability to fit any selected curve with a B-spline Curve with specified spline degree and number of control points.

The command is also available when a Wireframe entity is selected. It fits each chain in the .3DA or .PAT file with a B-spline Curve.

Cascade

Window>Cascade

Sizes and arranges windows as overlapping panes with all title bars visible and the active window on top.

To make the title bars of all SurfaceWorks windows visible:

Click  or choose **Window>Cascade**.

Children

See "Adopt Children" on page 133

See "Select Children" on page 210

See "To hide Parents or Children:" on page 179

See “To show Parents or Children:” on page 217 .

Class Filters

Contours	**Planes**	**Surfaces**
Curves	**Points**	**Triangular Meshes**
Graphs	**Knotlists**	**Relabels**
Composite Surfaces	**Wireframes**	**Reals**
	Snakes	**Invert Class Filters**

Constrains selection by entity class. When a filter is on (checked in the dialog or pushed in on the toolbar), it allows entities of that class to “get through,” i.e. be available for selection. Conversely, when the filter is off, entities of that class are not available for selection (if you try to select them, they just won’t select). You can turn on as many class filters as you want.

Filters reset to their default settings (all on) each time you open a model.

To filter selection by entity class, via the menus:

1 Choose **Select>Class Filters**.

2 **Check** the boxes of the classes you want to be able to select; **uncheck** the boxes of those you don’t want to be able to select.

3 Click **OK**.

Set All activates all the class filters (all kinds of entities can be selected).

Clear All deactivates all the class filters (no kind of entity can be selected). This in itself is not very useful, but if you wanted to turn on just one or two of the class filters, you might find it easier to Clear All, then turn on the one or two you want, rather than turning off individually all the ones you don’t want.

Invert turns off all currently set classes and turns on all the classes that weren’t set.

To filter selection by entity class, via the toolbar:

- Click the entity class toolbar button to its “pushed in” or “on” state (light gray background) to make those kinds of entities available for selection, the same as checking the class in the Class Filters dialog.
  or
- Click the entity class toolbar button to its “unpushed” or “out” state (darker gray background) to make those kinds of entities not available for selection, the same as unchecking the class in the Class Filters dialog.
  or

- Click the Invert Class Filters button to make all currently available classes of entity unavailable, and at the same time make all currently unavailable classes of entity available.

Clearance

Tools>Measure>Clearance

[Only available when a point and one other entity are selected, or a Fitted Surface is selected.]

Calculates and reports either minimum distance or quality of fit, depending on the entity(s) selected.

To calculate minimum distance:

1 Select a **point entity**.

2 Then **add another entity** to the Selection Set.

3 Click  or choose **Tools>Measure>Clearance**.

Minimum distance: SurfaceWorks calculates the distance between the point and the closest point on the curve, surface, or plane; or the distance between two points (this latter gives the same result as **Tools>Measure>Distance**). For a surface or plane, the distance can be positive or negative; a positive clearance means the specified point is along the positive normal direction.

When a curve or surface is chosen, SurfaceWorks also gives you the t-value or the u- and v-values for the point on the curve or surface closest to the point you are measuring.

To calculate quality of fit:

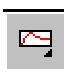
1 Select a **Fitted Surface**.

2 Click  or choose Tools>Measure>Clearance.

Quality of fit: SurfaceWorks compares the Fitted entity with its parent entity. It reports the number of control points used and the RMS (root-mean-squared) deviation of the fitted surface from the parent surface.

RMSN = normal component	RMST = tangential component
RMS = total deviation	log RMS = logarithm of RMS

Clearance Profile

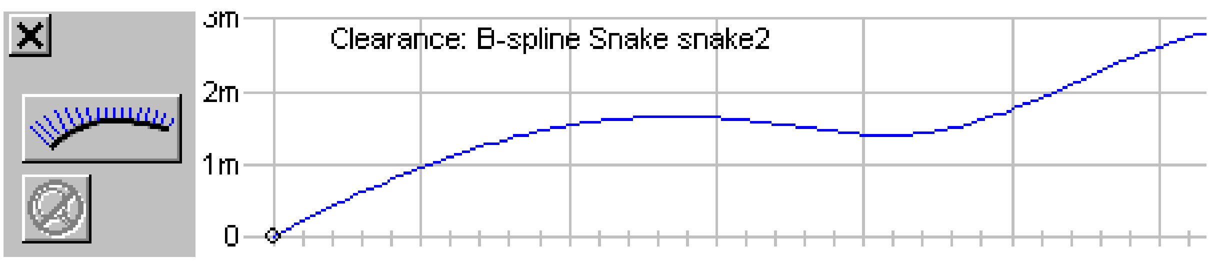


View>Display>Profiles>Clearance

In complex designs there are often needs to make two curves or surface edges that are as close as possible to identical, or a curve that lies on a plane or surface as accurately as possible. SurfaceWorks provides many constructions that allow such joins to be made exactly, often by means of various kinds of snakes. But sometimes it is necessary to approximate these joins; for example if your design is restricted to NURBS surfaces, which usually can't make exact joins, or must be communicated through an IGES file, which cannot represent the exact junctions.

One way to check the distance between a curve A and another curve B is to put an Bead 'e1' on A, at some arbitrary parameter value, and a Proximity Bead 'e2' on B, at minimum distance from 'e1'. Then with Tools/Distance or Tools/Clearance you can read the distance between the two beads (which would be zero if the junction were exact. Moving 'e1' to various positions along A and repeating Tools/Clearance would eventually let you assess how the clearance is distributed. If the question is clearance between a curve and a surface, you could do the same thing with a Proximity Magnet on the surface; if it's clearance between a curve and a surface edge, you can put an Edge Snake on the surface and use a Proximity Ring.

Clearance Profile automates this procedure and displays the results as a graph, similar to Curvature Profile and Tangency Profile. Clearance profile is enabled when there are two objects selected, the first being a curve (including a line or a snake) and the second being a curve, surface, or plane. The quantity displayed is a distance (units of length), vs. the parameter value t on the curve. Clearance from a plane or surface is a signed quantity, positive in the direction of the positive normal. Clearance from a curve is always positive or zero.



In this example, the selection set consists of B-Spline Snake 'snake2' on one surface, and the matching Edge Snake on an adjoining surface. The clearance is zero at the two ends. This model is dimensioned in meters, so the maximum discrepancy is approximately 3.0 mm, at about t = 0.68.

Clear Selection Set

Select>Clear Selection Set or **Ctrl+L**

[Only available when at least one entity is selected.]

Removes all entities from the Selection Set, leaving it empty.

To clear the Selection Set:

Click , or choose **Select>Clear Selection Set,** or press **Ctrl+L**, or click in whitespace in a graphics display.

Close

File>Close

[Not available in SolidWorks Integration mode.]

Closes the current model (all windows).

To close the current model:

Choose **File>Close**.

Command Window

Tools>Command Window or **W**

Allows the use of SurfaceWorks commands. See the chapter on commands for more information.

Component

File>Component

SurfaceWorks components provide the ability to paste groups of geometrically related entities (a “component”) into a model (the “host” model), without having to construct the component entity-by-entity. Component files have the extension .MC2.

Generally, you make a component by saving a selected portion of an existing SurfaceWorks model (the “source” model). When SurfaceWorks saves a component file, it includes all the component’s parents from the source model as hidden entities in the component file. That way, you can open a component using **File>Open** (set "**Files of type:**" to "**All Files (*.*)**"). You would do this to:

View the component in its native setting. **Show-Hide>Show All** or  to see its parents.

Edit the component. Edit the entities in the component just as you would edit the entities in a model. To save the edited component, select all the entities you want to be in the component (just as you would if you were saving the component out of a model file), then **File>Component>Save**. For the editing/ reselecting/ resaving process, you may find it helpful to show the component’s parents (1st generation or more).

Save component as a model file.

Show-Hide>Show All or  to see its parents. **File>Save As** to save the component as a model file (.MS2).

General component issues

Parents: A component essentially is a chunk taken out of a SurfaceWorks model. As such, its dependencies usually are incomplete (like a fish out of water) — it needs parent(s). The host model parent(s) must be akin to those in the source model. E.g. if the component is a C-spline Lofted Surface that had 3 master curve parents in its source model, the parents for the component in the host model must be 3 entities that qualify as curves (curves, snakes, points).

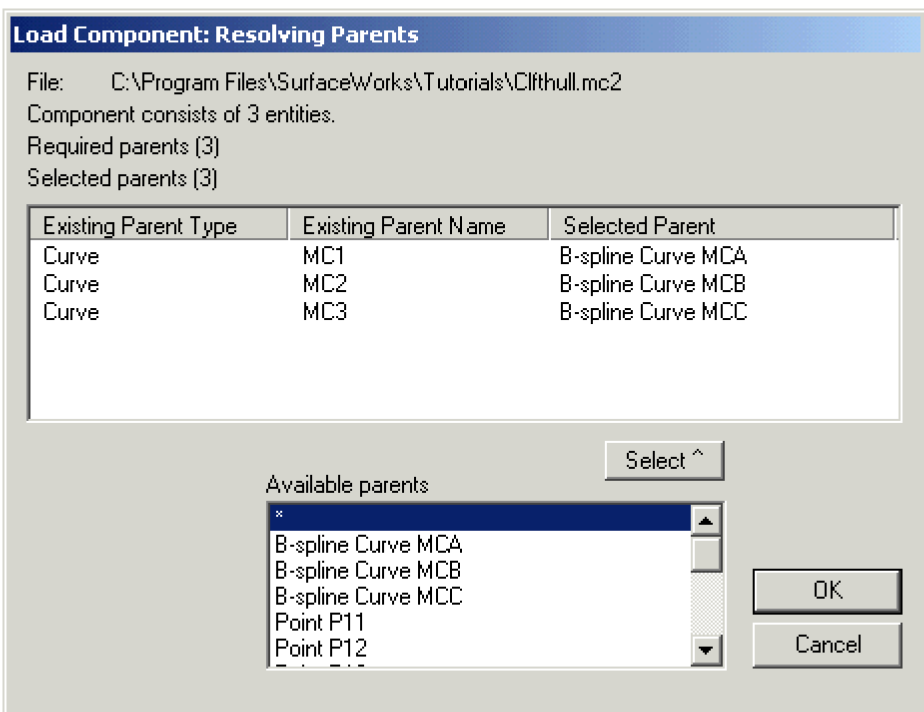
Orientation: Beyond kinship between source and host parents, there is the matter of matching orientation (t for curves and snakes; u,v and normal orientation for surfaces) between source and host. If a component's attachment to a host surface is with magnets, the host surface must have u,v running in the same direction as the source surface.

Load

Imports the contents of a SurfaceWorks component file (.MC2). For a component to work correctly, its parents and orientation must be compatible with the host model (see General component issues above and the examples in Tutorial "Working with Components"). You may preselect the required parents or select them in the Resolving Parents dialog. If you preselect parents, select them in the correct order (of course, if you miss, you can always reselect them in the dialog box). For examples of loading components, see Tutorial "Working with Components".

Resolving Parents dialog

Required parents. The number of parents required is shown after the heading. In the list box, the required parents are listed in order. Each entry states the kind of entity required (point, curve, surface, etc.). The name which follows is the name that parent had in the source model from which the component was saved. This name is included for reference only; the parent in the host model can have a completely different name. But note that if parents are not pre-selected and an appropriate parent of the same name exists, SurfaceWorks does the selection itself.



Selected parents. The number of parents selected is shown after the heading. When all the necessary parents have been selected, this number will be the same as the number of required parents. Each entry displays either *EMPTY* (no parent selected) or the entity type and the entity name of the parent chosen from the host model.

To specify a parent, highlight the parent in the "Selected Parent" list, then pick the parent using the “Available parents” list box. Either double click the parent or select the parent and click on <Select^>.

Available parents. This box lists the entities available as parents for the selected “Required parent” entity. To choose a parent, double-click it in the list, or highlight it and then click the <Select ^ > button.

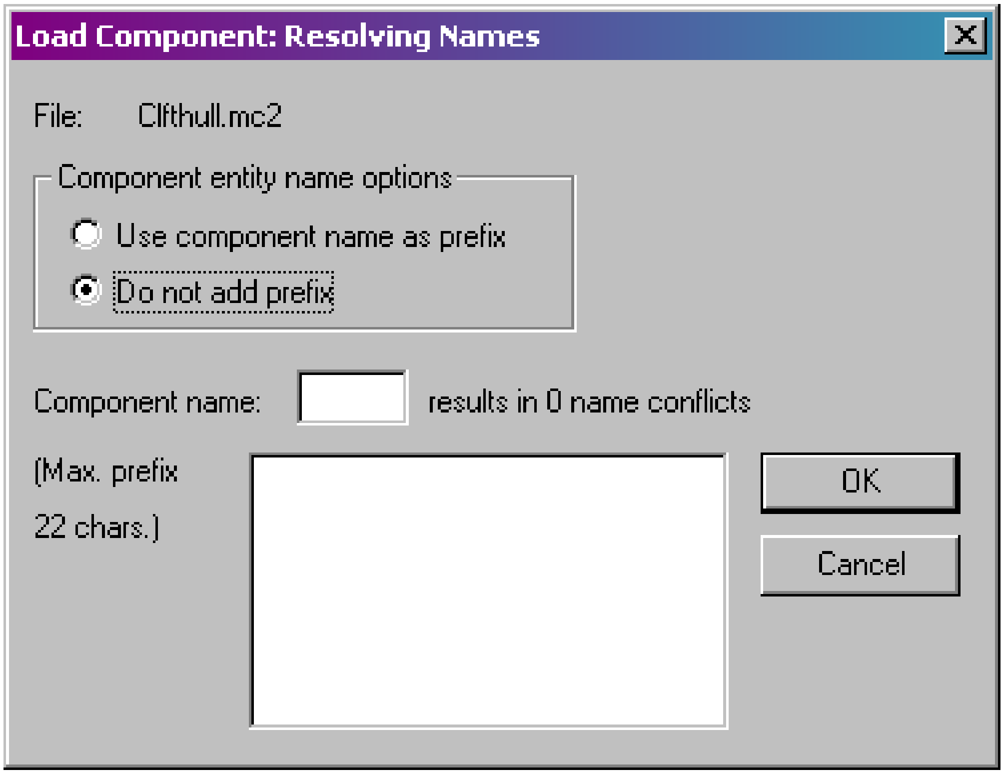
<OK> moves you on to the Resolving Names dialog.

<u>32-character entity names</u>: If the component you are loading contains one or more entities with 32-character names AND there are name conflicts, you will be notified that name conflict resolution cannot proceed. To fix the problem, you will need to: <OK> the message box, open either the component or the component’s source model, edit the names of one or more entities, and save the component again.

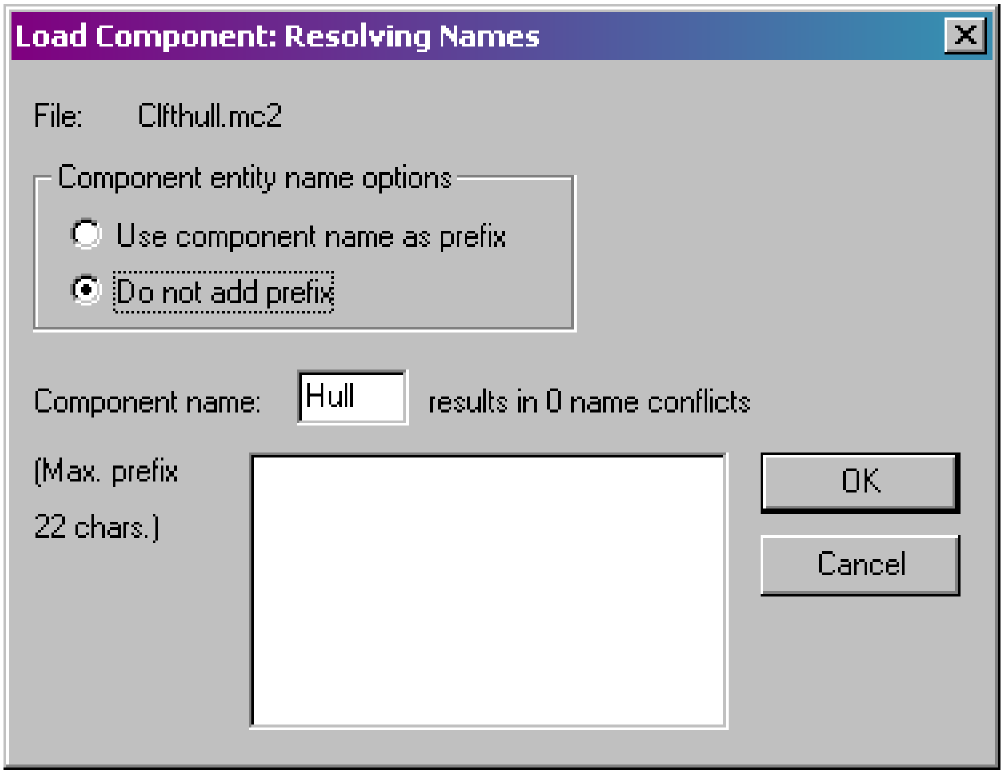
<Cancel> cancels the component loading process and returns you to the drawing.

Resolving Names dialog

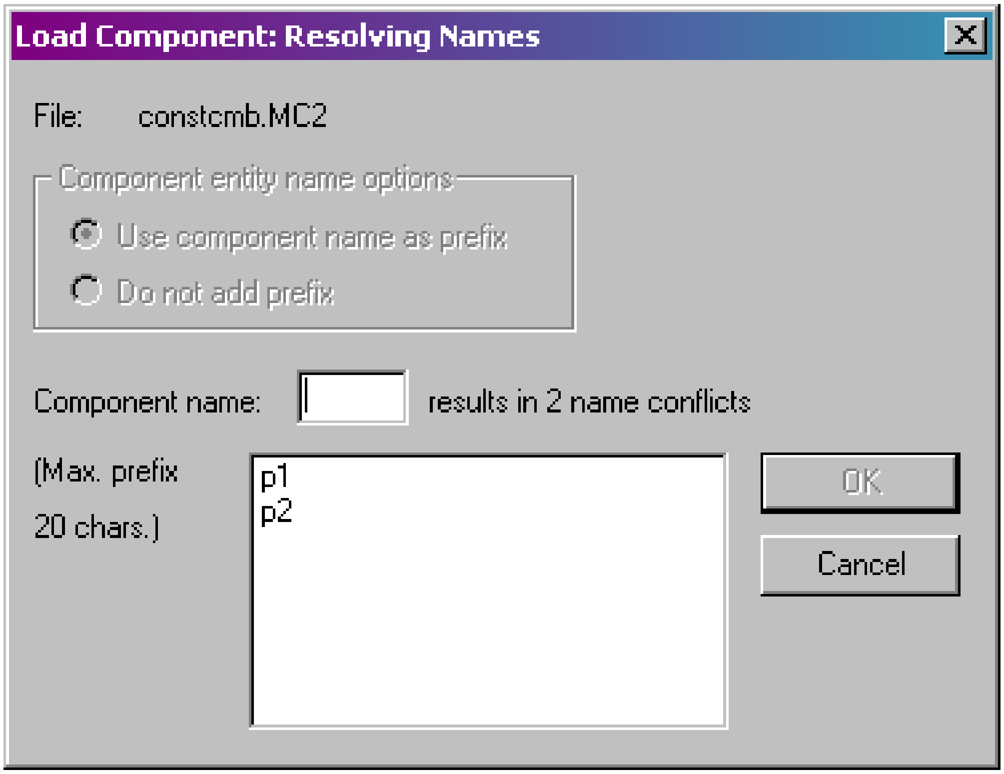
Prefix. *NEW* All component entities can have a prefix whether there is a name conflict or not. A prefix could give the user a big advantage in a large model to determine the structure and inter-connection of entities in a model. As shown in the first Resolving Names dialog below, the prefix field is blank. The field following reports the number of name conflicts which exist between the component entities and the host model entities. In this case there are no conflicts and a prefix would not be required. The radio button “Do not add prefix” is selected indication the user preference. To the left of the conflicts field is the notation of the maximum number of characters remaining available for the prefix.



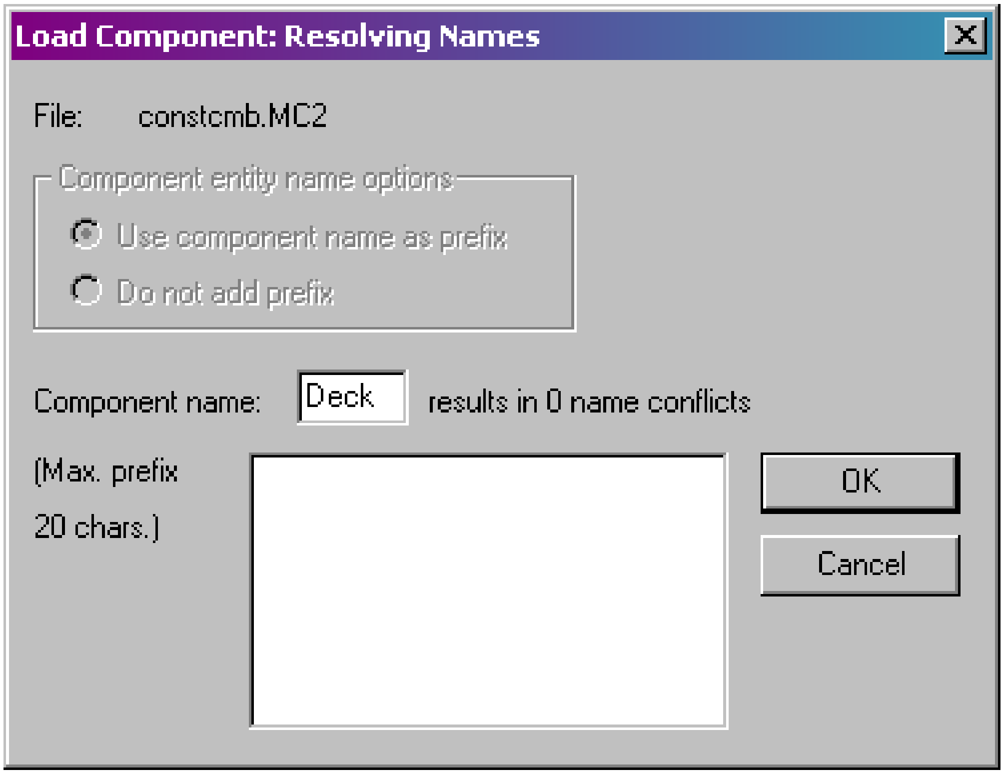
***NEW* Component Name** All components must have a name. They are now listed in the Entity Manager for enhanced editing. In the Resolving Names dialog below the required name has been added. All entities will be added to the model without a prefix and in the Entity Manager this component can be located under the Components heading as 'Hull". Should the user desire a prefix, the "Use component name as prefix" radio button would be checked. The component would be named "Hull" and all entities names, in the added component, would be similar to: "Hull.entity_name". MultiSurf now uses a period to separate component prefixes from entity names.



Resolving Name Conflicts In the first Resolving Names dialog example below we are loading a constant camber deck on to the hull previously loaded. There are two name conflicts and we no longer have the option of not using a prefix. The prefix is required to resolve the conflict. There cannot be two entities in the model with the same name. It is also noted that the prefix doubles as the component name.



The <OK> button will not be active until name conflicts have been resolved. To resolve name conflicts, type a prefix into the Prefix field. This prefix will be added to all component entity names. Here is an example:

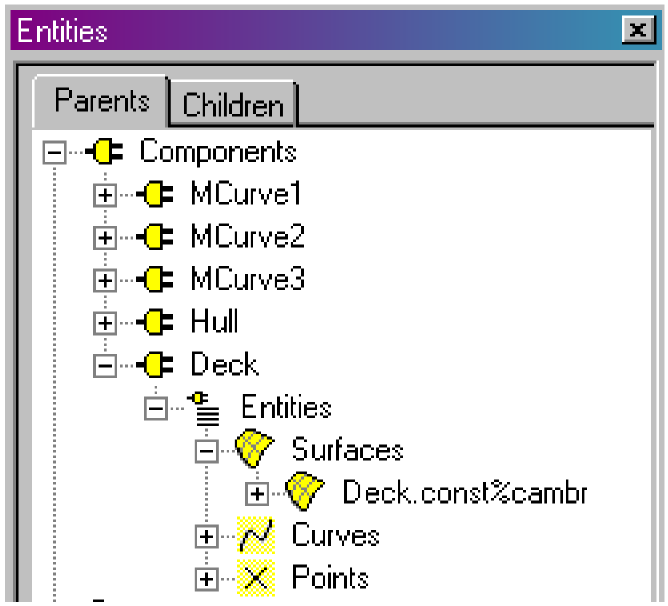


<OK> loads the component; the component's entities are all selected. This makes it easy to move all the component entities to a new layer, or to Delete the component if you don't like the result. You can also use Undo to remove the component if the Load result is tangled or confusing.

<Cancel> cancels the component loading process and returns you to the drawing.

(Note for those of you who look at or edit model files as text: the identifying message, entities, and remarks of the component appear at the end of the host model file.)

NEW The Component in the Entity Manager



New Component Heading The model we have been using as an example was created by loading multiple components which can now easily be seen by looking under the new Component Heading in the Entity Manager.

Each component is listed by name and arranged in a tree format similar to the model as a whole. These can be used as a model tree within a tree to separate components and simplify the understanding of complex models.

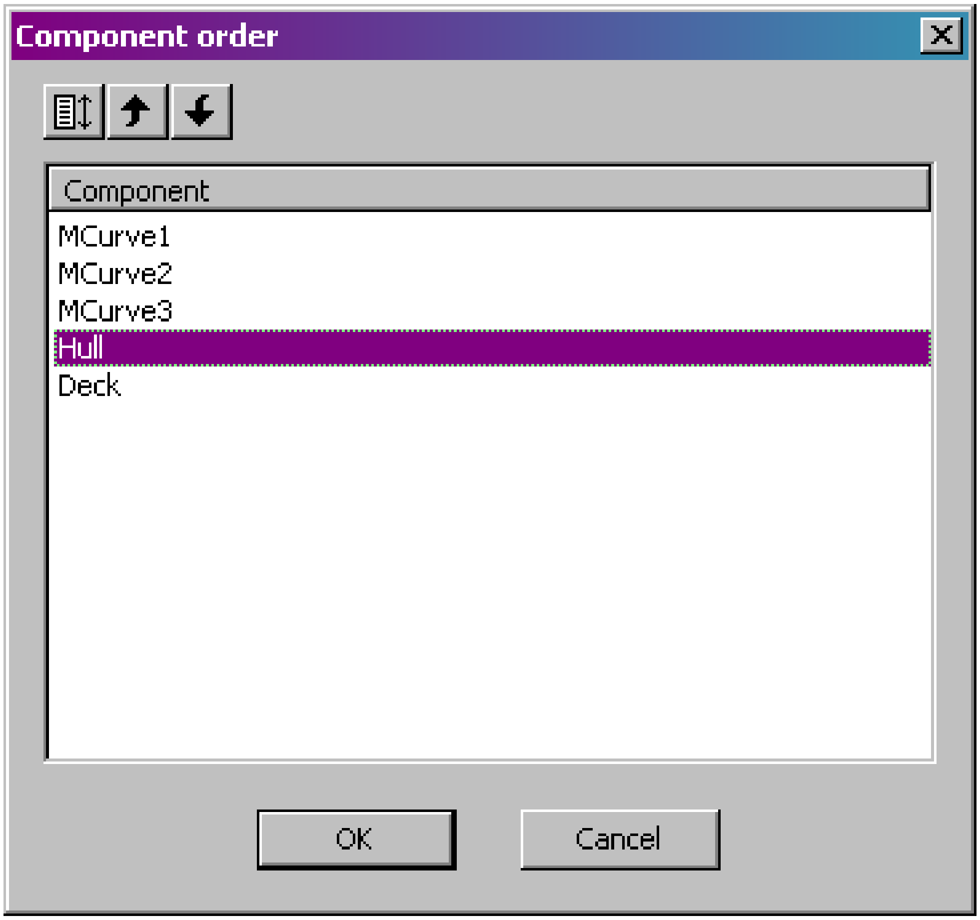
NEW Changing the Component Order

There are many enhancements added to the Entity Manager regarding the editing of components. It has been our intention to keep them apart from the model and easily influence the component as a whole with the use of the context sensitive (right click) menu.

Right click on the Component heading and two choices are provided:

Load – Load a component

***NEW* Change order** – Choose this and you will get the following dialog:



Select various components and use the arrow buttons to move the components to the desired order.

NEW Editing Individual Components in the Entity Manager

Right click on a component name in the Entity Manager and you will be presented with the following choices:

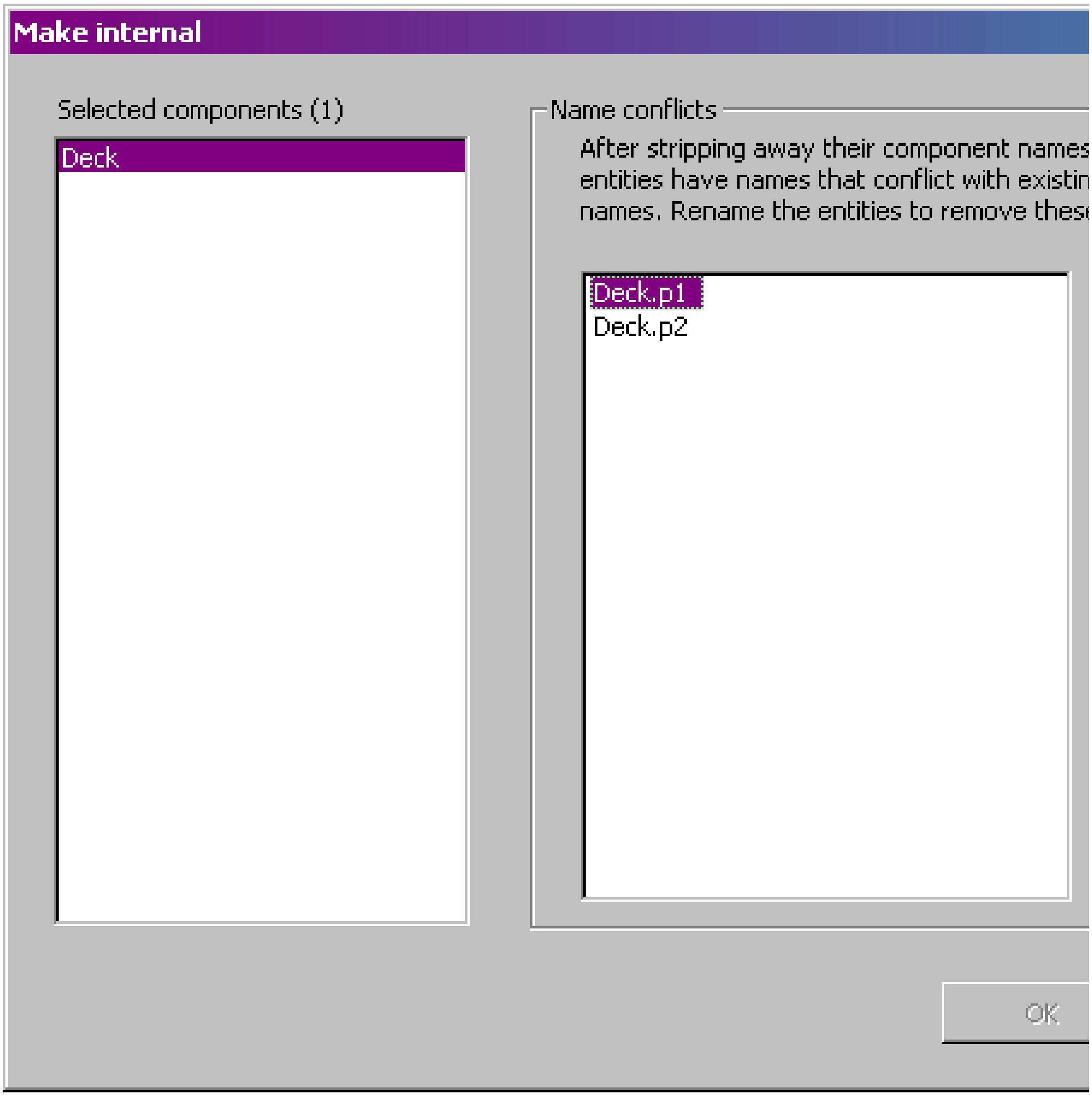
Select Selects all the entities in a component. Useful for query purposes, changing layers, and Multiple Edit.

Delete Deletes all component entities and removes it from the Entity Manager.

Show Changes the visibility state of all component entities to “visible”.

Hide Changes the visibility state of all component entities to “not visible”.

***NEW* Make Internal** At some point it may be desired to fold the contents of a component into the main body of the model. This could be advantageous when removing prefixes is needed or if a minor construction component was added for efficiency, but it does not need to be separated from the model with its own heading. We call this action Make Internal. During this action, any prefix is stripped and the component heading is removed. The action of stripping the prefix could result in name conflicts, which can be resolved in the following dialog:



***NEW* Current** A Current Component is similar to a Current Layer. All subsequent insertions of entities will be added to the Current Component.

Save

Saves a selected set of (nominally) related entities into a component file (.MC2). This component can then be loaded into another host model (as long as the host contains suitable parent entities for the component), or it can be opened to begin a new model or simply to view it in its native setting (**Show-Hide>Show All** or  to see its parents).

For examples of saving components, see Tutorial "Working with Components".

Note about component entity names: When you are saving a component, you will probably want to avoid entity names that are 32 characters long (the maximum entity name length). Because ... when you load a component into a host model, if there are any component entity names which are the same as host model entity names, you will be required to specify a "prefix" that will be added to <u>all</u> the component entity names. If any component entity names already are 32 characters long, adding a prefix will be impossible and you will have to go back and save your component anew, with different names.

Selecting the component entities

Whether you are saving a component from a model file (.MS2) or from a component file (.MC2) you opened and have edited, <u>before</u> choosing **Component>Save**, you need to select the entities that will comprise the component. In doing this, you will probably find (**Select>Parents>First Generation**) and (**Select>Children> First Generation**) useful. There are several ways you might do this:

Suppose, for example, that you want to make a component from a keel that is attached to a hull surface with magnets:

- You could select the keel surface(s), then do a series of (**Select>Parents>First Generation**) until the parent that would be added was the hull — you'd cancel from that one (because you don't want the hull in the component), then choose **Component>Save**.

- Or you could select the keel surfaces, then choose **Select>Parents>All Generations** — this would select more entities than you want, but you could then use to remove them from the Selection Set pane or <Ctrl>+click to get rid of the unwanted ones from the graphic view(yes, in a complex model, this could get messy).

- Or … you could select the magnet(s) that attach the keel to the hull, then choose **Select>Children>All Generations**.

- If you use a relabel entity, be sure to check dependencies in <u>both</u> directions (parents and children), because these entities often are little side branches in the dependency tree (and as well, they aren't visible).

- Or, of course, you could select the entities one-by-one in the drawing — but that could be very tedious, and you'd run the risk of missing hidden entities and non-visible entities such as Relabels this way (oops!).

You can use the Selection Set pane to see (and change if need be) a list of the entities you've selected to include in the component.

Saving the component file

With the component entities selected, it's time to select File>Component>Save.

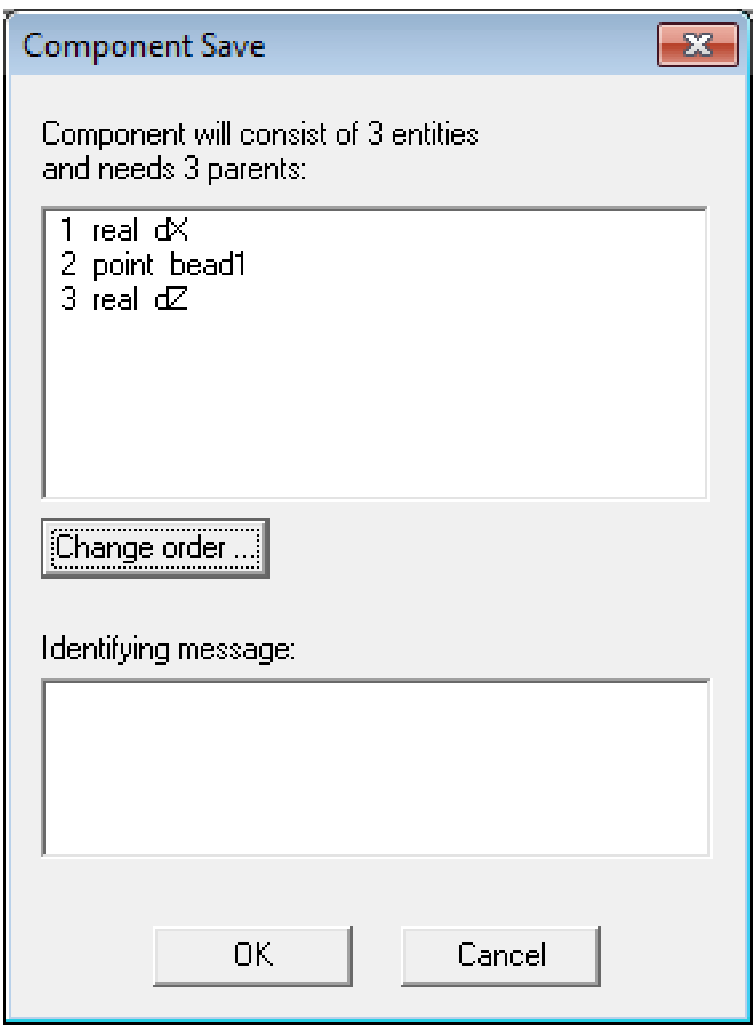
Parent(s). This list displays the kinds of parents the component will require when it is loaded into another model. The entity name is the name of the parent in the current or "source" model (that is, the model from which you are saving the component). It is listed as a way of helping you keep track of connections. The parents in any future host model are not required to (although they may) have the same parent names as in the source model.

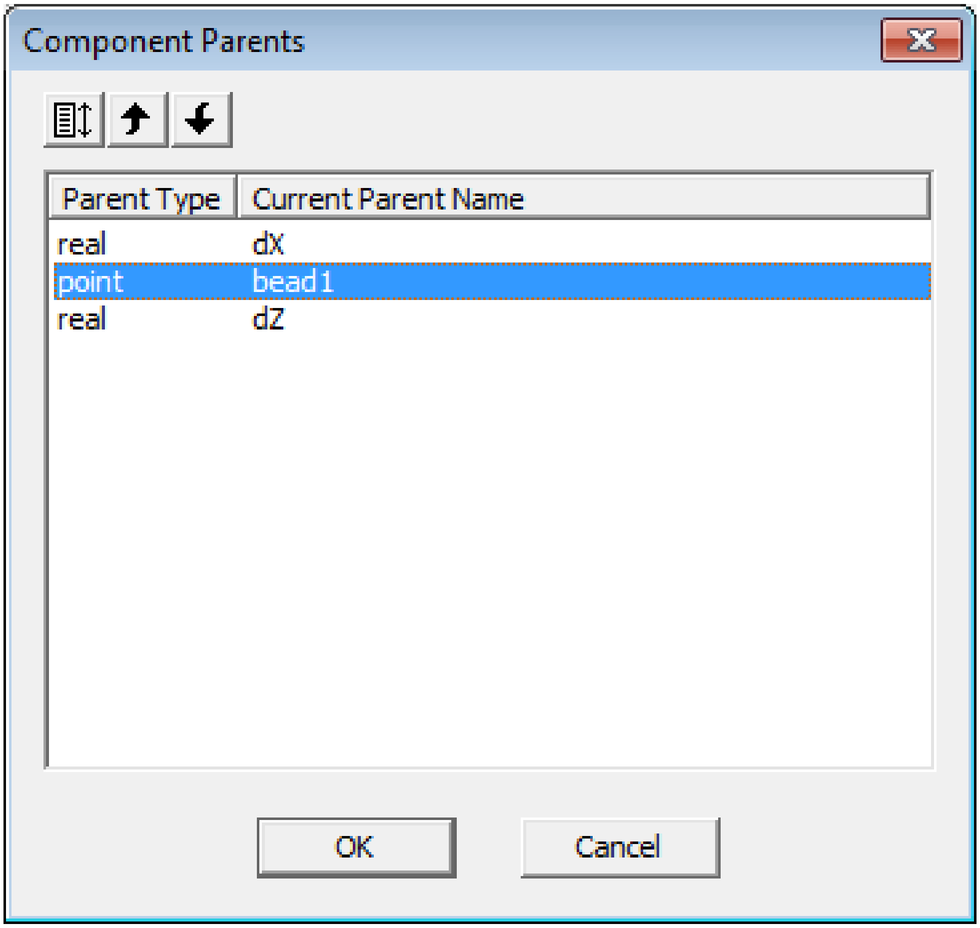
***NEW* Change Order Button** This is used in the case the list of Parents are in an inconvenient order for re-insertion into a model. Please look at the images below to see the list of Parents required for insertion. For this component the ‘bead’ parent will be the only one changed and the two variables will be taken from already existing entities. The bead should be at the top of the list. For pre-selection of parents the best solution is to select a bead, and on Component / Load the new bead will be the only entity which needs to be selected. The two variables will automatically be added to the Resolving Parents list.

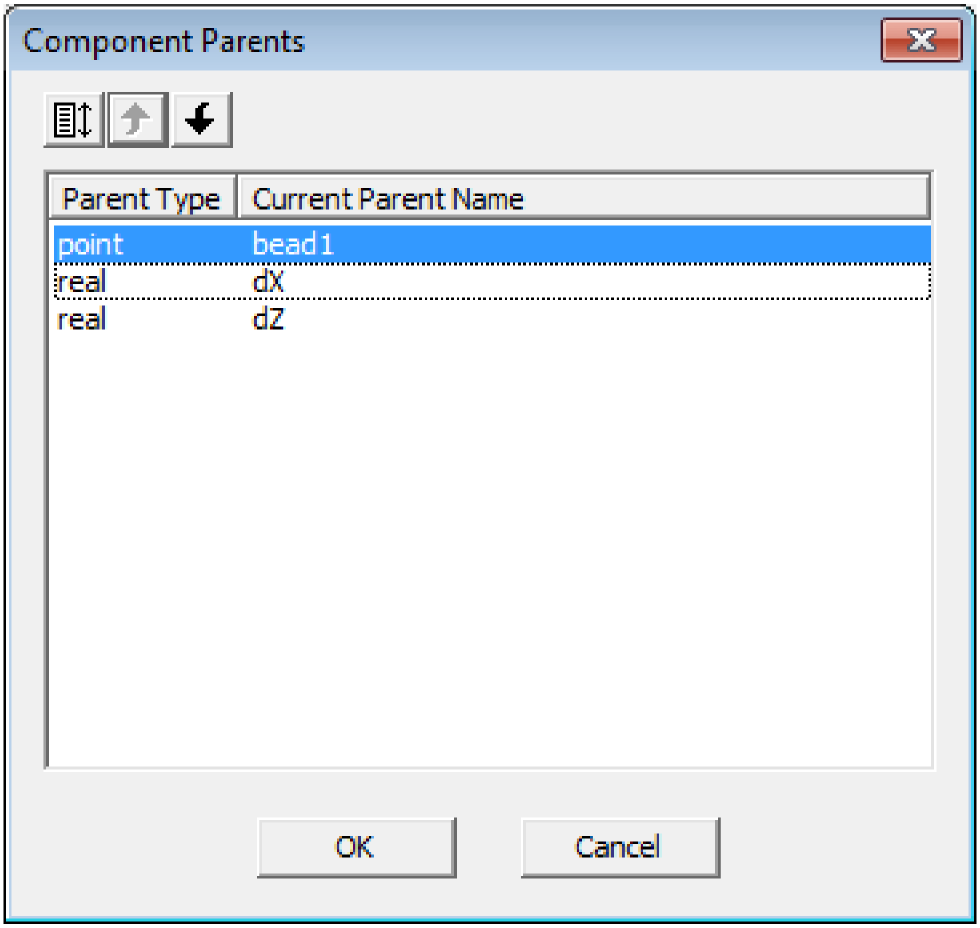
Identifying message. Type in an identifying message for the component. This message will be put in the heading of the component file and inserted into any host models when the component is loaded into them.

<OK> takes you on to the Save Component dialog where you can specify a filename (and path) for the component. You will be warned if any component entity names are 16 characters long.

<Cancel> returns you to the drawing with the Selection Set intact.




Component Parents
Parent Type
Current Parent Name
real
dX
point
bead1
real
dZ
OK
Cancel

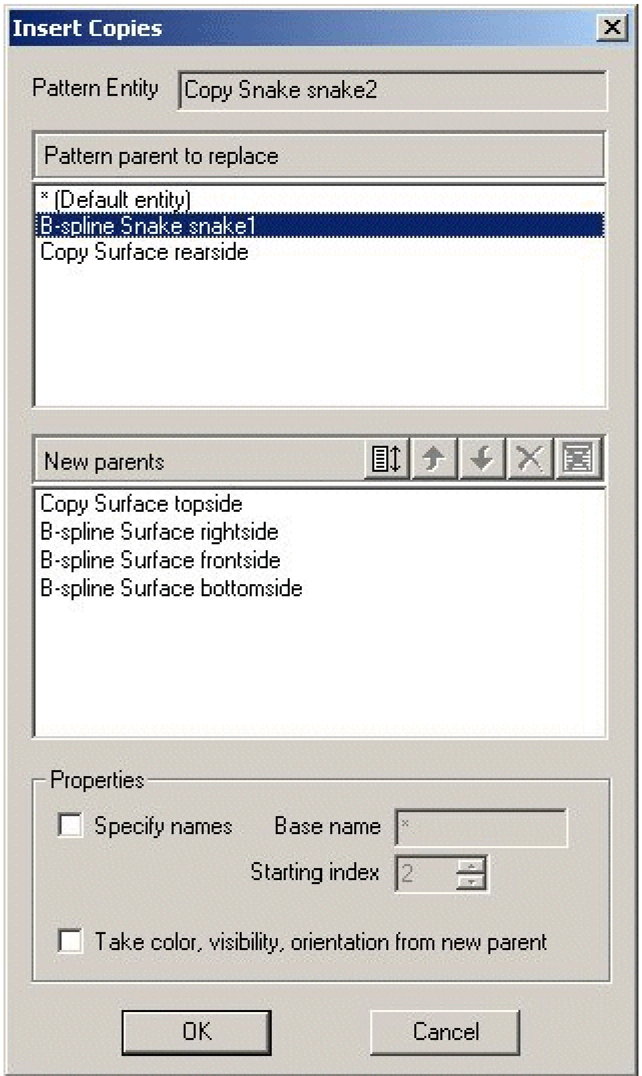


Copies

Insert>Copies is a SurfaceWorks tool for "mass production" of entities that have similar constructions. You basically say, "I want to create a bunch of entities that are like this one (the same entity type, in particular), but constructed using this and this and this, etc., as supports".

See CreateCopies in the Commands chapter.

You select the entity to be copied (the "pattern" entitity) first, then the set of new supports, and pick Insert>Copies from the menu. (or the Keyboard Shortcut <C>) This opens the Insert Copies dialog, as follows:



Patern Entity

The "Pattern entity" window shows the name of the pattern entity -- the first entity selected, in this case a Copy Snake. Each of the copies will be a Copy Snake.

Pattern parent to replace

The upper list box, "Pattern parent to replace", shows all the parents of the pattern entitity, in order. The one that is highlighted is the one that will be replaced in making the copies. You can click on a different parent to change the selection. (In this case, since the "New parents" are snakes, B-spline Snake snake1 is the only parent that makes sense for replacement; the new parents cannot serve as either relabels or surfaces.)

New parents

The lower list box, "New parents", shows the remainder of the selection set. If there are more entities selected than can be displayed, the window will have a vertical scroll bar; you can scroll up and down to review the list. In this case there are 3 new parents, so we are making 3 copies.

In case, on reviewing the "new parents" list, you find you have included one or more entities that shouldn't be there, you can highlight the incorrect entity and click the "Remove" button, , to remove it from the set.

Specify names

The "Specify names" checkbox allows control over the naming of the copies. Names are constructed from a base name by adding indices; you can choose the base name and the starting index. In this case the copies will be named 'D2', 'D3' and 'D4'. (If any of these names is already in use, the dialog will inform you and allow other choices.)

If you leave "Specify names" unchecked, the program will choose an available base name for you, and use 2 as the starting index.

Take color...

The final checkbox "Take color, visibility, orientation from new parent" transfers these attributes from the new parent. (The alternative is to take them from the pattern entity.)

Copy Last

Insert>Copy Last or **F4**

[Only available after at least one insert dialog has been opened during the SurfaceWorks session.]

Brings up the Insert entity dialog for a new entity of the same kind as the last insert dialog opened in the model. Defaults provided during **Copy Last** depend on the contents of the Selection Set when **Copy Last** is invoked (for details about the defaults, see ”Creating Entities - Introducing the Insert Dialogs - Defaults”.

Note: Neither Quick Points nor Quick Splines affect Copy Last.

To create an entity of the last type created:

1. If necessary, modify the contents of the **Selection Set**, to set up defaults for the copy as you desire.
2. Choose **Insert>Copy Last** or press **F4**.

Current Layer Filter

Select>Current Layer Filter

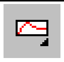
Constrains selection to entities on the current layer only. The menu option is a toggle, checked when on.

Current Layer Filter resets to off each time a model is opened.

To constrain selection to entities on the current layer only:

Click or choose **Select>Current Layer Filter**.

Curvature Profile



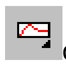
View>Display>Profiles>Curvature

[Only available when an appropriate curve is selected. The following curves are not appropriate: Mirrored Curves, Projected Curves, Relative Curves, and any type of snake.]

Displays a graph of curvature vs. arc length for the selected curve entity (the entity must be visible). Curvature graphs are an important tool for checking the detailed shape and fairness of a curve. The curvature profile graph is displayed in a window attached to the lower edge of the graphics window.

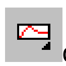
To display a curvature profile graph:

1 Select an appropriate **curve** entity.

2 Click  or choose **View>Display>Curvature Profile**.

To see the curvature profile for another curve:

1 Select the new entity.

2 Click  or choose **View>Display>Curvature Profile** again.

To see more detail in the graph:

Sliding the scroll bar along the right-hand edge of the curvature display in an upward direction magnifies the graph. This action is like zooming in, but only the vertical aspect is changed. This feature is particularly useful when you want to see more detail in the areas of the curvature profile that are near zero.

To turn off the curvature display:

Click the **Close** button in the graph window.

The Graph

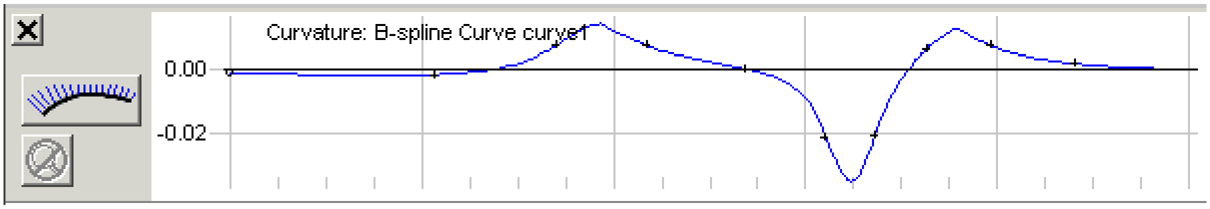
In the curvature profile graph, arc length (measured from the t=0 end of the curve) is the horizontal axis and curvature is vertical. The origin of the graph can be at either the right or left end of the screen, but it is always marked by a circle. The graph has tickmarks that correspond to the tickmarks you can display on the curve (visibility = curve and tickmarks); you may find it useful to edit the curve's visibility and turn them on. The curvature displayed is that of the current projection of the curve; consequently, the profile changes as you change the view.

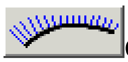
Curvature is the inverse of the radius of curvature (a tight curve has high curvature, a straight portion has zero curvature). The sign is significant. A zero crossing of the horizontal axis in the curvature profile is an inflection point on the curve.

For a brief discussion of reading the curvature profile display and some examples, see “Editing Models - Fairing Curves - Curvature Profiles: Basic Reading”.

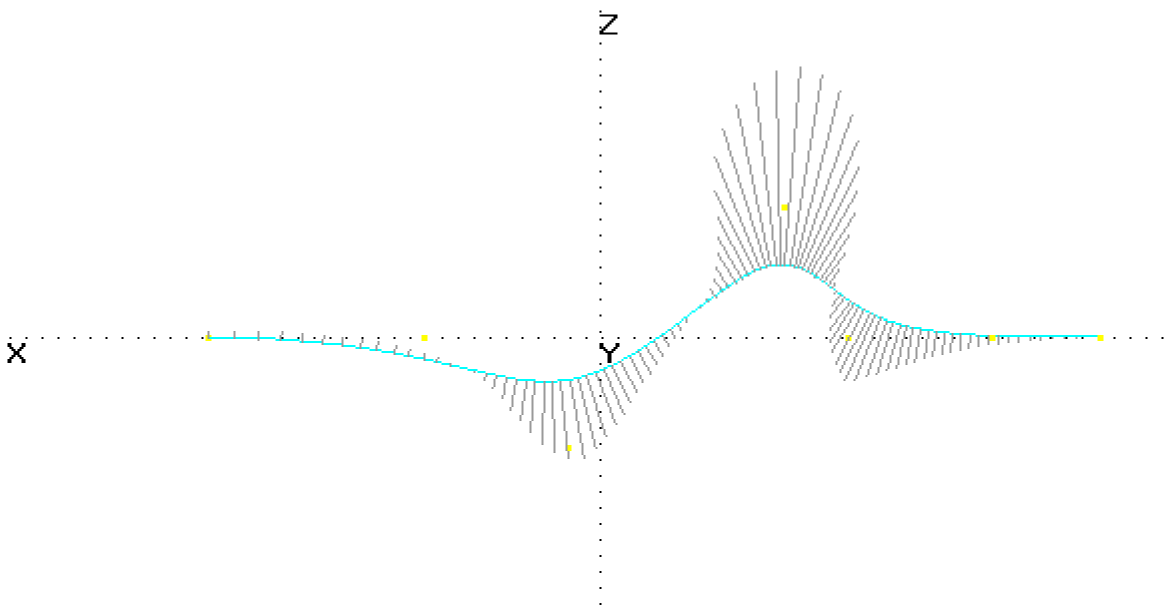
For information about turning the curvature profile display on/off, see “Curvature Profile” on page 156.

Porcupine



Clicking on the button to the left (seen above in the Curvature Profile view) will enable an alternate method of displaying a curves curvature profile. This is commonly called Porcupine Curvature Display. The spines of the curve represent the amount and direction of curvature. See below for an example from the model view.

Another new feature in this view is the ability to display the curvature profile of Snakes. Previously only curves were eligible.



Decimal Places

See "Performance Options" on page 198

Delete (Edit Menu)

Edit>Delete or **Delete**

[Only available when at least one entity is selected.]

The function of the Delete key or Edit>Delete varies depending on the contents of the Selection Set:

If an entity is highlighted in the Selection Set pane, SurfaceWorks will try to delete that one entity.

If there is not a highlighted entity in the Selection Set pane, SurfaceWorks will try to delete the entire contents of the Selection Set.

An entity can be deleted only if either it has no children or all generations of its children will also be deleted at the same time, i.e. all generations of its children are also in the Selection Set.

To delete one or more entities from the model:

1 **Select** an entity you want to delete.

2 Choose **Select>Children>All Generations** to add all its children to the Selection Set. (Read the list of entities that will be added to the Selection Set. Ask: Do I really want to delete all of these?)

3 **Repeat** the above steps for any additional entities you want to delete.

4 Press the **Delete** key or choose **Edit>Delete**.

If there are children outside the Selection Set:

If any entity to be deleted has one or more children outside of the Selection Set, the program will tell you and will not delete any entities.

To reverse the deletion of a set of entities:

Click  or choose **Edit>Undo**.

Display Options

Tools>Options>Display

Lets you set shaded and wireframe display options: shaded entity colors and degree of transparency, shaded mesh line display, and background color for shaded and wireframe windows.

All of the settings on this tab are saved with the application.

To set display options for shaded and wireframe windows:

1 Choose **Tools>Options>Display**.

2 Make your changes, then click **OK**.

Shaded:

Background color. Click the **Color** button to display the standard Windows palette. Choose one of the basic colors, or define your own custom background color; the program default is gray. You can also vary hue, saturation, and luminosity settings for custom colors.

Entity Colors. Choices are “Wireframe colors” or “Monochrome” (radio buttons); if monochrome, specify the color by choosing it from the drop down color palette. The program default is “Wireframe colors.”

Transparency. Use the slider to make the shaded image more transparent (i.e. lighter; move slider toward the right) or less transparent (i.e. darker; move slider toward the left). The initial program default is 50%.

Mesh lines. Check box toggles display of surface u,v mesh lines. The initial program default for mesh lines is on.

When mesh lines are on, the mesh lines displayed will be whatever is set in each surface's definition. The kind of lines displayed is set in the entity's Insert/Edit dialog, on the Display tab, under Visibility — there are check boxes for boundary, u-constant lines, and v-constant lines; the default when a surface is created is u- and v-constant lines on. The number of mesh lines displayed is controlled by the first value for u-divisions and for v-divisions on the Advanced tab.

Note: Trimmed Surface exception. When you turn off the mesh line display for the shaded image, Trimmed Surface mesh lines still will display. If you want Trimmed Surface mesh lines not to display, Edit Definition for the surface, turn on the boundary, then turn off the u- and v-constant lines.

Note: In a surface's definition, turning off the boundary and both u- and v-constant lines is the same as hiding the entity — the surface will not display (in this case, mesh lines on/off is immaterial). In order to avoid confusion about the visibility of an entity, if you try to turn off Boundary, u-constant and v-constant for an entity, the program will automatically set u-constant and v-constant 'on' (checked) and check Hidden in the Insert/Edit dialog. You can turn off the boundary and u, v lines for a Trimmed Surface as long as the triangles are on. It will not be checked as hidden.

Wireframe:

Background color. Click the **Color** button to display the standard Windows palette. Choose one of the basic colors, or define your own custom background color; the initial program default is gray. You can also vary hue, saturation, and luminosity settings for custom colors.

Distance

Tools>Measure>Distance

Provides a dynamic display of the Euclidean (3-dimensional) distance and the dX, dY, dZ distance between two selected points. Under appropriate conditions where the two points lie on the same curve or snake, Distance also reports the arc length distance between the two points. The report values will update dynamically when an associated point is moved. You can reposition the report displays manually.

To use Distance Mode:

1 Click or choose **Tools>Measure>Distance**. If Tools>View Distance Reports was off, it will automatically be toggled on. When you move the pointer back into the graphics window, it will display as .

Note: You may preselect two entities before you enter Distance Mode. See step 2 for guidelines. When you preselect entities and then choose Distance Mode, the report box will appear when the pointer is back in the graphics window.

2 To measure distance, **select two point entities**.

To measure arc length, **select two point entities** — at least one of the selected points must be a bead or a ring, and the other point must be either a bead or a ring on the same curve or snake, or must be an end point of the curve or snake. Arc length is a signed quantity, positive in the direction of increasing t.

A report-box outline will appear.

3 **Drag the report-box outline** where you want to position it, then **click the mouse**. The distance report will appear in the box.

4 If you'd like, **select two more points and position a report box for them**. You can do this for as many pairs of points as you'd like.

5 To exit the Distance mode, **right-click and choose another mode** (e.g. Select Mode), or press **Esc**, or click  again, or choose **Tools>Measure>Distance** again.

Note: Distance reports are not persistent between SurfaceWorks sessions. They also are not included in Undo/Redo sequences.

To reposition a report box:

1 **With the cursor pointed inside a report box, press down the left mouse button**. The pointer changes to .

2 **Drag the report box** to a new location and **release** the mouse button.

To turn on/off the display of distance reports:

1 Choose **Tools>Measure>View Distance Reports**. This menu option is an on/off toggle. When it is preceded by a check mark, all existing reports will be displayed; when there is no check mark, the display is off. The default when you start SurfaceWorks is on (checked).

2 To change (toggle) the display status, just **click the menu item**.

Note: If the View Distance Reports option is off when you enter Distance mode, it automatically will be toggled to on.

To permanently remove individual distance reports:

1 **Right-click in a report box**, then click **Remove**.

2 **Repeat** for any other reports you would like to turn off.

Divisions Multiplier

See "Performance Options" on page 198.

Dragging Options

Tools>Options>Dragging

 **Set All Snaps**  **Clear All Snaps**

 **Tools>Options>Dragging>Orthogonal** or **F7**

Lets you specify the settings for the dragging aids: snaps, orthogonal dragging mode, automatic display of Location dialog after drag and nudge movement amount.

In SurfaceWorks, snaps constrain point movement during dragging (in 3D space, along a curve or snake, or on a surface). You can set the precision of the snaps. Snap increments are invisible in the graphics window but the snapped-to values are displayed dynamically on the Status Bar.

Note: Snap is different from the lined grid (**View>Grid**) you can display in the graphics windows when in axial orthographic views (such as the standard views with perspective turned off).

The initial program defaults are all snaps toggled on with **X, Y, Z values** set to 0.01 and **t, u, v values** set to 0.001; **Orthogonal Dragging** and **Prompt after drag** are off and **Nudge amount** is 0.1.

Snap values and the **Nudge amount** are saved with the model. **Snap on/off settings** and **Prompt after drag** are saved with the application. **Orthogonal Drag** is reset to off each time you open a model.

To specify dragging settings:

1. Choose **Tools>Options**.
2. Click the **Dragging** tab.
3. Make whatever settings you need.
4. Click **OK**.

<u>**Snap**</u>**:**

X, Y, Z, t, u, v snaps. Toggle on / off (check boxes) and set increments (as integer or decimal values) for any combination of X, Y, Z, t, u, v snaps. You might think of the X, Y, and Z snaps as an imaginary 3D grid of points located the snap distances from each other; the snap-grid is parallel to and has the same origin as the world coordinate system. The dragged Point will snap to locations in the snap-grid. On curves and snakes, the snaps are at t-values and on surfaces they are at u,v-values. In these cases, the constrained point (Bead, Magnet, Ring) snaps to the parameter values, except for XYZ Beads and XYZ Rings, which snap to X, Y, or Z values.

The minimum allowed value for X, Y and Z snaps is the same as the value in **Tools>Options>Performance>Decimal places**. The minimum value for t, u and v snaps is .000001 (10E-06).

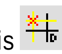
The snap values do not all have to be the same — e.g. the set of snaps X-snap = 5.35, Y-Snap = 0.01, Z-snap = 0.05 ... is okay (although probably not very useful).

Set All sets all snaps to on. The toolbar button is .

Clear All sets all snaps to off. The toolbar button is .

Dragging:

Orthogonal Drag. Toggles Orthogonal Dragging mode on / off (check box). When Orthogonal Dragging mode is on, dragging movement is constrained to being parallel to one or the other of the screen axes. The point will “land” where the cursor hairs cross (this may not be where the cursor itself is located).

The toolbar button for toggling Orthogonal Dragging is ; the shortcut key is **F7**.

You probably will find orthogonal dragging most useful in the six standard (axial) views (Front, Back, Left, Right, Top, Bottom) (and Bow, Stern, Port, Starboard, Deck, Keel if you have Marine) with Perspective turned off. In these cases, two of the world coordinate axes are parallel to the two screen axes, therefore only one coordinate will change as you drag — the other two will be held steady. (Visual cue: in the 6 axial (standard) orthographic views, when the grid is on, both horizontal and vertical grid lines are displayed.)

You also can use orthogonal drag in orthographic views that have only one world axis parallel to a screen axis. The results are somewhat more difficult to predict by eye because two coordinates will change as you drag — and one will be held steady. (Visual cue: in these orthographic views, when the grid is on, there are either horizontal or vertical grid lines, but not both.)

When you use orthogonal drag in other orthographic views or in perspective views, the results can be quite difficult to predict by eye. (Visual cue: in these views, even when the grid is on, no grid lines are displayed.)

Prompt after drag. Set (check box) whether or not you want the program to automatically display the Location dialog box when you drag a point. The Location dialog box lets you adjust the point’s location by typing in one or more of the location values (and requires an **OK** or **Cancel** response in order to continue).

Nudge amount:

The value is multiplied by the **Model Units** length value. If **Model Units** is Inches / Pounds, for example, and **Nudge amount** is .1, the distance a point is moved for each keystroke is 0.1 inches.

Edit Menu

See “Undo” on page 227.
See “Redo” on page 206.
See “Delete (Edit Menu)” on page 157.
See “Edit Model File” on page 163
Transform>Scale See “Scale” on page 225.
Transform>Shift See “Shift” on page 226.
Transform>Rotate See “Rotate” on page 226.

Edit Model File

Edit>Model File

With this menu selection you can view or edit the current model file as text, using the mouse and/or the keyboard. Typing overwrites selected text or inserts at location of cursor.

Warning

You can do a lot of damage using Edit/ Model File, because the editor has no knowledge of the proper contents and format of a model file. There is no assurance that your edited model file will evaluate correctly, unless you conform completely to the file specification. By contrast, during other editing operations within the program, SurfaceWorks is responsible for maintaining the consistency of the data, and (provided we have done our job correctly) will never produce an invalid file.

Entity Options

Tools>Options>Entity

Lets you specify settings for certain entity tools. Currently you can set Quick Spline Mode defaults and IGES Export tolerance setting.

The entities on this tab are application settings.

To change any of these settings:

1. Choose **Tools>Options>Entity** tab.
2. Make your changes.
3. Click **OK**.

Quick Spline Defaults:

Choose to create **B-spline Curves** or **C-spline Curves** in **Quick Spline Mode** by clicking the appropriate radio button. Default is B-spline.

Degree. Specify the degree of spline ((1 = linear, 2 = quadratic, 3 = cubic, etc.) by putting the appropriate number in the degree box. Default is 3.

Alternate Spline Creation. Specifies the user interface behavior when creating Quick Splines. When this is checked, click the mouse button at the location of each control point, except double-click to drop the last control point (this ends the spline). See "Creating Entities - Quick Points and Quick Splines - Spline Creation Method Options".

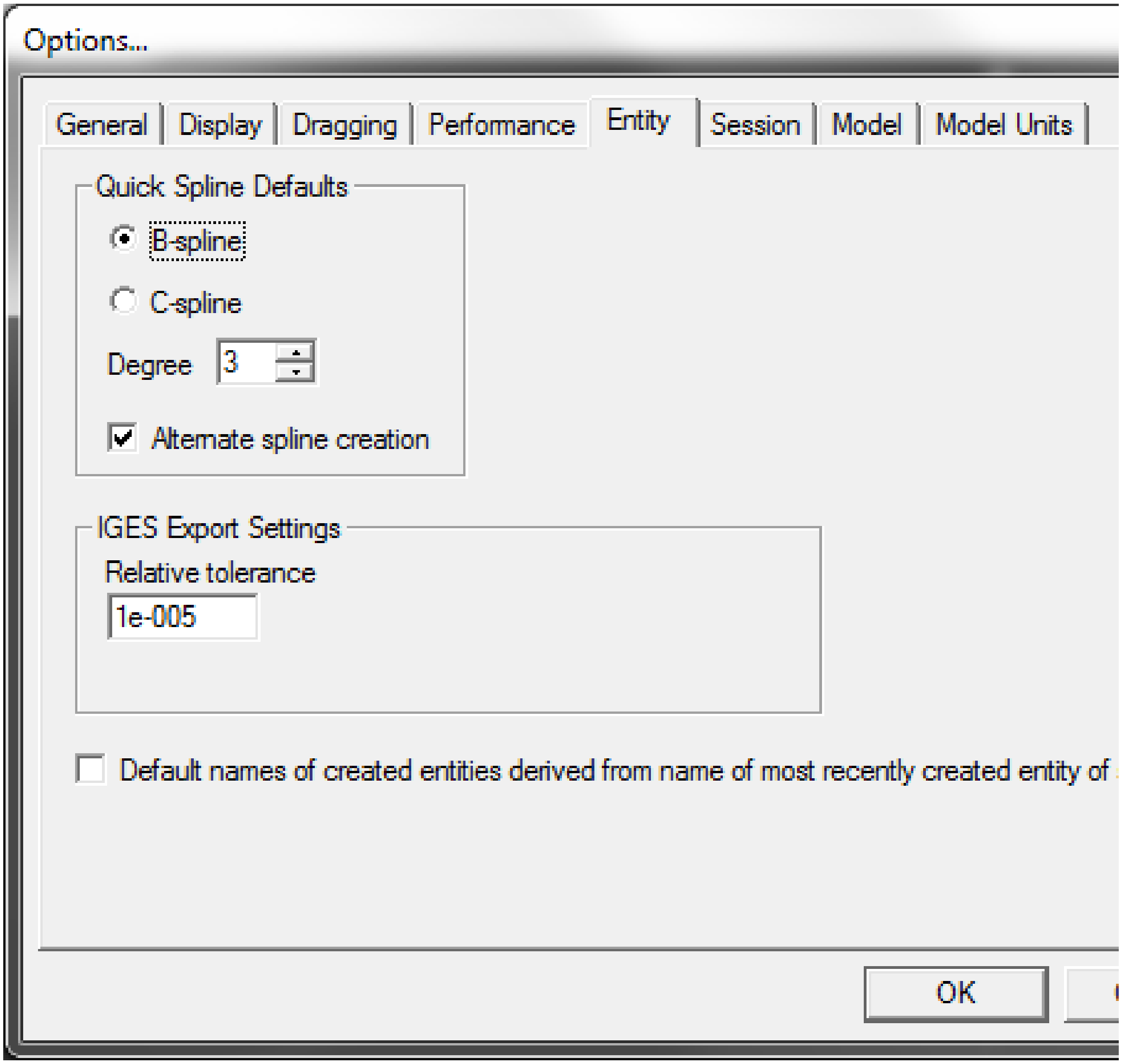
IGES Export Settings:

The default tolerance is based on the general size of the entire model. Loosening the tolerance allows a fit with fewer control points, but may result in poor approximation. Tightening the tolerance may improve the approximation, but will result in more control points. More control points results in larger files and slower performance in the importing system.

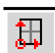
New Naming Convention Option

In the Entity Tab of the Tools/Options dialog there is a new check box. Here you are able to choose one of two naming conventions. The default option is a new scheme which DOES NOT use the names of previously named entities, of the same class, to use as a basis for newly created entities.

Should you choose to keep the same naming convention always used by /SurfaceWorks, check the box to keep the traditional naming techniques.



Entity Orientation



View>Entity Orientation

[Only available when a single curve or surface entity is selected.]

Pins the display of parameter orientation information to a curve, snake, or surface entity. This user-specified orientation display is a toggle that remains on, no matter what is in the Selection Set, until you turn it off by choosing **View>Entity**

Orientation again or until you select a different entity and choose **View>Entity Orientation** to pin the orientation display on the new entity.

This setting resets to no entity selected when a model is opened.

Note: If you wanted to have SurfaceWorks automatically display orientation marks for single or all selected entities — the display shifting as the Selection Set changes — you'd use the automated Auto Orientation option (see "Auto Orientation" on page 135).

To pin the display of parameter orientation to an entity:

1 Select a **curve**, **snake**, or **surface** entity.

2 Click  or choose **View>Entity Orientation**.

The pinned orientation display for an entity remains until you turn it off by:

- Choosing **View>Entity Orientation** again.
- Selecting a different entity and choosing **View>Entity Orientation** to pin the orientation display to the new entity.

The Entity Orientation Display

Parameter orientation is displayed in two ways, depending on what kind of entity is selected:

Curves, snakes: A circle is drawn around the t=0 end of the curve; and an arrowhead is drawn halfway along the curve, pointing in the direction of increasing t.

Surfaces: A circle is drawn around the u=0, v=0 corner of the surface; a "u" (for u-direction) and an arrowhead are displayed halfway along the v=0 edge, and a v (for v-direction) and an arrowhead are displayed halfway along the u=0 edge. An arrow, emanating from the center of the surface, points in the direction of the positive normal to the surface. On Trimmed Surfaces only the positive normal is displayed.

The orientation marks are displayed in black.
Auto orientation marks see "Auto Orientation" on page 135"
Orientation Mark options see "Tools Options" on page 224"

Errors Window

See "What's Wrong" on page 234.

Exit

File>Exit

[**Exit & Return** in SolidWorks Integration mode.]

Ends the current session of SurfaceWorks and closes the program window.

To end the current session:

Choose **File>Exit**.

Export 3DA wireframe

File>Export 3D>3DA Wireframe

AeroHydro's ASCII 3D wireframe format; similar to a DXF file in that it is a file of colored line segments. Among other purposes, you might use such a file to create a WireFrame entity copy of an original model to load with a modified model, for comparison.

A .3DA file contains wireframe and / or panel information in ASCII format. It consists of an unlimited number of records each in the format:

pen% x y z

pen% = integer controlling pen color; 0 = pen up
x, y, z = world coordinates of point

Spaces can be used for separators.

A panel is represented in a .3DA file by a group of 3 or more records having the following special characteristics:

pen% = 0 for the first point
pen% > 0 for other points
x, y, z of first and last point are identical

To export surfaces for .3DA:

1 **Select** the entities you want to export **or make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 3D>3DA wireframe**.

3 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Entities** or **Selection Set Only.**

5 **Click the check box** to **Include Symmetry images** if desired; **OK.**

6 **Specify a filename** for the .3DA file; **Save**.

Export DXF

File>Export 3D>DXF and **File>Export 2D>DXF**

Export 3D>DXF

DXF format is AutoCAD's ASCII "drawing exchange format," which can be read directly into AutoCAD, and is accepted as a de facto standard by many other CAD programs. This format presents curves as polylines and surfaces as either a set of polylines or a mesh representation consisting of a set of 4-sided panels.

To export surfaces for 3D DXF:

1 **Select** the entities you want to export **or make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 3D>DXF**.

3 **Click the radio button** to export **All Visible Entities** or **Selection Set Only.**

4 **Click the check boxes** for **Points**, **Curves**, **Surfaces** and **Contours** as desired to modify your choice in step 3 as indicated. A check mark indicates that that entity type will be exported. If not checked, that entity type will not be exported.

5 **Click the check box** to **Include Symmetry images** if desired.

6 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

7 **Click the radio button** to choose how you want the export to deal with layers.

- **Export all entities to single layer** is self explanatory.
- **Use model layers** puts the entities on the layer that they appear in the SurfaceWorks file
- **Entities on individual layers** puts each entity on its own layer.
- **Entities and each contour cut on individual layers** places each entity on its own layer and also places each contour cut on its own layer. For example, if you export 3 surfaces and a 10 station contour, in addition to a layer for each of your 3 surfaces, you will have 10 layers each containing one of the 10 contour cuts.

8 **Click the radio button** to export curves, isoparms, contours, arcs and lines as **polylines** or as **AutoCAD SPLINE, ARC and LINE entities**.

9 **Click the radio button** to export surfaces as **isoparms** or as **polygon meshes**.

10 **Click the radio button** to choose your file format: **AutoCAD R12/LT2 DXF** or **AutoCAD 2000 DXF**. **OK**.

11 **Specify a filename** for the DXF file; **Save**.

Export 2D>DXF

The current screen view of all visible entities, or of the current Selection Set, will be captured as a 2D wireframe for transmission to a CAD drafting program.

To export entities for 2D DXF:

1 **Select** the entities you want to export **or make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 2D>DXF**.

3 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Objects** or **Selection Set Only.**

5 **Set Point size** to the desired size for the exported points. In 2D DXF, points are exported as octagons. **Point size**, which can be any number, determines the size of these octagons. The default of 1 makes the points the same relative size as they appear on the screen. **Point size** = 2 makes the points twice as large as **Point size** = 1, etc. **Point size** = 0 draws the points as single dots (which usually makes them invisible in the picture).

6 Write an **Identifying Message** if you would like. <**OK**>**.**

7 **Specify a filename and directory** for the DXF file; **Save**.

Condensing polylines in 2D DXF export

Curves, snakes, surface edges and contours are all output as polylines in 2-D DXF exports. Frequently, one of these elements consists of a lot of short segments that all add up to a line. For example, think of a station cut on a boat hull in a plan or profile view. The contour data it's drawn from is a 3-D polyline with (typically) 10 to 100 segments. In the past, this would result in a 2-D polyline with the same number of segments.

Now, each polyline that's projected is first subjected to a test to see if its projection adds up to a single straight line in the current view. If so, it's "condensed" for output as a single line. This results in a much "lighter" drawing, that is much faster to display and easier to modify in the drafting program.

Export IGES

File>Export 3D>IGES

Makes a B-spline approximation of all the visible or just the selected surface entities in a SurfaceWorks model, output in IGES format as NURBS Surface entities (entity 128) and Trimmed Parametric Surface entities (entity 144). Also supported are the Point (Entity 116), Composite Curve (Entity 102), NURBS Curve (Entity 126) and the Curve on a Parametric surface (Entity 142) entities.

Note: For both accuracy and speed of approximation for IGES export, it is helpful if the divisions in your model are coordinated. Coordination should include the divisions between surfaces and their parent curves, as well as the divisions of adjacent surfaces that have edge continuity imposed. For information about how to coordinate divisions, see "Understanding SurfaceWorks Entities - Defining Entities - Additional Surface Properties - Coordinating Divisions".

To export surfaces for IGES:

1 **Select** the entities you want to export **or make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 3D>IGES**.

3 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Objects** or **Selection Set Only.**

5 **Click the check box** to **Include Symmetry images** if desired; **OK.**

6 **Specify a filename** for the IGES file; **Save**.

7 Since there are inherent inaccuracies in the fitting process, we strongly recommend that you **view the results of your export**. You can use any 3D program which imports the IGES format, including SurfaceWorks.

If the Export Results Are Not Satisfactory

If the results of an IGES export are not satisfactory, you have a few of options:

- Check your model for division coordination. If need be, increase divisions on surfaces in question to maintain coordination with parent curves, then re-export from SurfaceWorks.
- Re-export from SurfaceWorks using a different tolerance. To change tolerance use **Tools>Options>Entities.**
- As a "last resort": make B-spline Fitted Surfaces of your SurfaceWorks surfaces before exporting IGES. This gives you the greatest control over the accuracy of the exported surfaces because you have specific control over the number of control points used as well as of the individual surface tolerance. For details about Fitted Surfaces, see "Entity Descriptions - Fitted Surface".

Poor Candidates for IGES Export

A few surfaces that can be created in SurfaceWorks are not well approximated by B-spline surfaces. Example:

- Any surface that has a crease (e.g. a lofted surface formed by one or more master curves with discontinuous slope) will be poorly approximated. Such a surface should be split along the crease into two SubSurfaces, each of which is then smooth and suitable for B-spline approximation.

How Exported Trimmed Surfaces Are Represented

Trimmed surfaces are stored in the IGES file as IGES type 144 Trimmed Parametric Surface.

The Trimmed Parametric Surface is defined as an IGES type 128 NURBS Surface (same way an untrimmed surface is stored) in combination with an IGES type 102 Composite Curve.

The Composite Curve is further composed of one or more IGES type 142 Curve on a Parametric Surface representing the trimming loops.

Finally, each Curve on a Parametric Surface is composed of one or more IGES type 126 NURBS Curve(s) representing individual trimming snakes.

How SurfaceWorks Generates Exported Surfaces

Compared with surface meshes (as stored in a .DXF file), a B-spline representation is very compact and defines a smooth, continuous surface representation. Most MultiSurf/SurfaceWorks surfaces can be approximated quite accurately by a B-spline surface having a modest number of control points. The accuracy tolerance is specified by the user. To perform the fit, SurfaceWorks automatically steps up the number of control points until either the fitted surface falls within tolerance or the program limit on number of control points is reached. In this latter case, SurfaceWorks issues a warning that the tolerance was not met, but it outputs the surface anyway (because we suspect this is more useful than not). The default tolerance is based on the general size of the entire model.

Factors that affect the quality of the surface approximation are:

- The number of divisions on the original SurfaceWorks surface
- The variation of curvature across the surface — some surfaces which have large variations in curvature may be approximated poorly

Special case output: Some surfaces have an exact NURBS representation; for example, B-spline Lofted Surfaces that have all B-spline Curve master curves of the same type and the same number of control points. In such cases, SurfaceWorks just outputs the control points (it does not do a fitting).

Export PAT Meshes

File>Export 3D>PAT...

Exports a .PAT file from a selection Set of from the visible surfaces.

Export PMARC

File>Export 3D>PMARC

This export option will only be active with a PMARC Crypkey authorization.

Exports PMARC .geom and / or .wake files from a Selection Set or from the visible surfaces. PMARC is a panel method for analysis of aerodynamic flows past complex configurations, developed by NASA at Ames Research Center.

Export POV triangles

File>Export 3D>POV triangles

A POV-Ray format file consisting of triangular entities. POV-Ray makes high quality rendered images - and it's free. Unfortunately, with recent more-complex models, we have been discovering how very slow it can be at the higher resolutions we want to use. Still, it makes very nice pictures, and SurfaceWorks makes POV-Ray input files directly.

POV-Ray setup and procedural considerations

You create an input file for POV-Ray using **File>Export 3D>POV triangles**. This option saves a file containing all currently shown surfaces in your model (including symmetry images if you turn on the checkbox in the Options dialog) in the format required for POV-Ray. Each surface object or image is turned into a POV object consisting of a union of smooth triangle entities. The triangles are made by diagonally cutting each quadrilateral in the nu x nv surface mesh (where nu = u-divisions x u-subdivisions, nv = v-divisions x v-subdivisions).

POV files are very bulky, approximately 230 bytes per triangle. For example, the DEMO model, with 10 x 20 subdivision, generates 400 triangles and approximately 92K of POV data, with symmetry images NOT included. This is a fairly coarse subdivision for wireframe purposes, but it appears quite smooth in POV-Ray. You may need to output a few surfaces at a time, in order to make POV files that are small enough to permit editing of features such as lighting sources, surface colors and textures. You can do this by hiding surfaces. Following editing of individual files, the whole POV model can be assembled from the parts by using a text editor.

POV-Ray book and ordering

POV-Ray is copyrighted freeware created by the POV Team. We recommend an excellent book on POV-Ray: Ray Tracing Creations, Second Edition published by The Waite Group (200 Tamal Plaza, Corte Madera, CA 94925 USA); phone 800-368-9369 (USA) or 415-924-2576; FAX 415-924-2576. The book includes a disk with version 2.2 of POV-Ray. You also can download version 2.2 from the Internet or CompuServe, or you can have us send it to you.

To export surfaces as .POV triangles:

1. **Select** the entities you want to export **<u>or</u> make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2. Go to **File>Export 3D>POV triangles**.

3. **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Objects** or **Selection Set Only**.

5 **Click the check box** to **Include Symmetry images** if desired; **OK.**

6 **Specify a filename** for the POV file; **Save**.

Export RUL Surfaces

File>Export 3D>RUL surfaces

[Only available if there are one or more rulable surfaces in the model]

AeroHydro's ruling file format for input into the MSDEV plate development utility. To qualify as a rulable surface, a surface must have straight parametric lines in either the u or v direction. Since not all rulable surfaces are developable, SurfaceWorks will notify you if a surface is not in fact developable (within a tolerance), but it will still allow you to save the surface in the RUL file — for details, see "Making input RUL files - RUL does not guarantee developability" in your MSDEV documentation.

The RUL file includes all visible contours, snakes, magnets, and rings on the selected surfaces. The orientation attribute of surfaces is important to MSDEV — for details, see "Making input RUL files - The orientation factor" in your MSDEV documentation. (See also "Appendix - Output File Specifications - RUL-Ruling File".)

To export surfaces for RUL:

1 **Select** the entities you want to export **or make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 3D>RUL Surfaces**.

3 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Objects** or **Selection Set Only**; **OK.**

5 **Specify a filename** for the RUL file; **Save**.

Export STL

File>Export 3D>STL

An STL (stereolithography) file is an industry-standard format for transmitting triangle mesh surfaces. Officially, the triangle mesh is supposed to enclose a solid, but the format works just as well, and is widely used, for triangle meshes that are not closed.

File/ Export3D/ STL will be enabled if (1) the selection set contains any entities that qualify for triangle export, or (2) the selection set is empty and the model has one or more visible entities that qualify. Currently, the entity types that qualify are:

all surface types including Trimmed Surface and Composite Surface

all TriMesh types
all solid types

An untrimmed surface or a solid face is divided into parameter-space rectangles along division and subdivision lines, then each rectangle is split along one diagonal into 2 triangles.

The STL file specification is actually two formats, one binary and one ASCII. The binary format is much more compact (about 1/4 the size) and is generally preferred. The ASCII format can be read (and, in principle, edited) in a text editor such as Notepad.

Export VRML

File>Export 3D>VRML

VRML Surfaces — VRML is an open standard for transmitting 3D scenes and environments over the Web. VRML is a textual language that describes 3D objects, their positions, lighting and other visual effects. VRML files are intended to be displayed using VRML browsers, which allow the user to navigate through the scene and interact with it.

SurfaceWorks exports the specified (visible or selected) surface objects as IndexedFaceSet nodes.

The specification for VRML files may be obtained from the following web site:

http://sunee.uwaterloo.ca/~broehl/vrml

The following reference includes the VRML 1.1 specification and code for a VRML reader:

Rule, Keith: 3D Graphics File Formats: A Programmer's Reference, Addison Wesley Developers Press, Reading, Mass (1996) ISBN 0-201-48835-3.

To export surfaces for VRML:

1 **Select** the entities you want to export **<u>or</u> make visible** all, and only, the entities you want to export. When you are exporting the Selection Set, it can include hidden entities and those on layers which are turned off.

2 Go to **File>Export 3D>VRML**.

3 **Click the radio button** to save in **Fast** or **Accurate** mode. Click the **Stay in mode after exit** check box if you changed the mode and want SurfaceWorks to stay in that mode after exporting. This change affects all exporting until reset.

4 **Click the radio button** to export **All Visible Objects** or **Selection Set Only.**

5 **Click the check box** to **Include Symmetry images** if desired; **OK.**

6 **Specify a filename** for the VRML file; **Save**.

Export WAMIT

File>Export 3D>WAMIT>GDF

File>Export 3D>WAMIT>FRC

These export options will only be active with a WAMIT Crypkey authorization.

Exports a .GDF file from a Selection Set or from the visible surfaces, or an FRC file from selected points, curves, surfaces and / or solids.

WAMIT is a radiation / diffraction panel program developed at M.I.T. for the linear analysis of the interaction of ocean waves with offshore structures. For more information, visit www.wamit.com.

File Menu

See "New File" on page 196.
See “Open” on page 196.
Import See "Import DXF" on page 179
See “Import IGES” on page 181
See "Import Triangle mesh" on page 186
See “Import .SLDCRV” on page 184.
Import MS2 (SolidWorks Integration only) See "Using SurfaceWorks with SolidWorks - Reference Guide to SurfaceWorks for SolidWorks Functions in SurfaceWorks - Import .MS2"
See “Save” on page 209.
See “Save As” on page 209.
Export 3D See “Export 3D>DXF” on page 167.
See “Export IGES” on page 168.
See “Export VRML” on page 173.
See “Export POV triangles” on page 171.
See “Export PAT Meshes” on page 170.
See "”Export PMARC” on page 170.
See “Export RUL Surfaces” on page 172.
See "Export 3DA wireframe" on page 166
See “Export STL” on page 172.
See “Export WAMIT” on page 174.
Export MS2 (SolidWorks Integration only) See "Using SurfaceWorks with SolidWorks - Reference Guide to SurfaceWorks for SolidWorks Functions in SurfaceWorks - Export .MS2"
Export 2D See “Export 2D>DXF” on page 168.
See "Close" on page 140
See "Exit" on page 165
Exit & Return (SolidWorks Integration only) See "Using SurfaceWorks with SolidWorks - Reference Guide to SurfaceWorks for SolidWorks Functions in SurfaceWorks - Exit & Return to ____.SLDPRT"

Filter

See "Class Filters" on page 137.

See "Current Layer Filter" on page 155.

See "Reset All Filters" on page 208

First Generation

See “Select Children” on page 210.

See “Select Parents” on page 212.

Flatten

Tools>Flatten

Only available if you have purchased our compound-curved surface flattening add-in, SurfaceWorks Flattener. Otherwise, this option will always be grayed out.

If you have SurfaceWorks Flattener, this option will be available when you have one or more surfaces or Triangular Meshes or Composite Surfaces selected. See the Flattener chapter on Page 16-1.

General Options

Tools>Options>General

In General Options, you set whether SurfaceWorks should start maximized, which sounds you want, whether you want nametags to be transparent, which Auto Orientation option you want, which rotation mode you will use and digitizer settings.

These settings are saved with the application i.e. SurfaceWorks will always open with these settings.

To set SurfaceWorks to start up maximized and to toggle sounds on/off:

1. Use the **Tools>Options>General** tab.
2. Make your settings, then click **OK**.

Startup Settings:

A checkbox toggles “Startup maximized” on/off. When checked, the program starts as a maximized window. The program default is on.

Sounds:

Checkboxes toggle sounds on/off for errors, filter failures (if you click on an entity, but the entity cannot be selected because it doesn’t pass current filter criteria), and when the program has finished drawing the image. Program default is all sounds off.

To have sounds, your computer must support sound and you must have Windows sounds turned on (e.g. go to **Start>Settings>Control Panel>Sounds** and under Events choose the **Window>Exclamation** option).

Nametags:

Check the **Transparent nametags** checkbox if you want nametags to be visible through one another in the graphics displays**.** If you want the nametags of the entities closest to you to be opaque, therefore not obscured by other nametags “behind” them, uncheck the box.

__Auto Orientation__:

None. Auto display is turned off. Orientation marks are not automatically displayed when an entity is selected.

Single selected entity. When the selection is a single entity, its orientation marks are displayed. This is the initial program default.

All selected entities. Orientation marks are displayed for all selected entities, no matter how many entities are selected.

__Rotation Constraint__**:**

Free tumble. The angle that each of the three axes makes with the vertical is allowed to change.

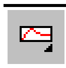
Z axis vertical. The z-axis is always vertical relative to the screen.

__Digitizer__**:**

Baud Rate. The default baud rate for the digitizer is 9600. This can be set to as low as 300 and as high as 38400. If you change settings, the new setting will remain between sessions.

Com Port. Your computer searches your available ports and lists them here. The port you use for the digitizer must match the port picked in the list box.

Graph Profile



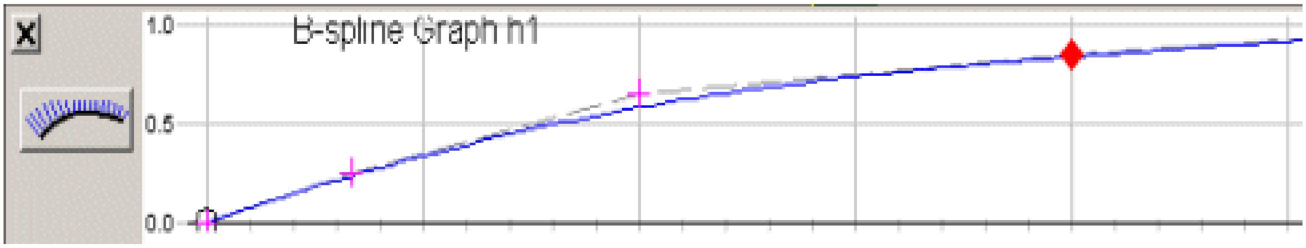
View>Display>Profiles>Graph

[Available when one Graph or Relabel object is selected]
This new Tools option provides a graphical interface for viewing Graph and Relabel objects.

A Graph object is a real-valued function of a single parameter (usually designated t, like the t parameter on curves and snakes). Graphs are used to greatly enrich the possible geometries created by several entities, for example Relative Curve, Offset Curve, Tangent Boundary Surface and Sweep Surface. Relabels are a special type of graph used to control the t parameter distribution on curves and snakes. Graphs and Relabels are non-graphic entities that can't be seen directly in a wireframe or rendered view (although they usually do have visible effects on the objects they support).

Note that as non-graphic objects, Graphs and Relabels can't be selected directly in the wireframe view. If you know their names, you can select them from Entities Manager, or with Select / By Name. Alternatively, select a graphic object that uses the Graph or Relabel as a support, and edit its Graph or Relabel attribute; then open the Graph Profile.

View / Display / Profiles / Graph opens a window similar to the Curvature or Velocity Profile windows, showing a plot of the graph over the parameter range 0 to 1, and the location of its control values.



This example is the graph used as a blending function in the sample file RELCURV1.MS2. It is used in the model to control the width of the spray flat on a planing hull bottom. It is a type-3 BGraph with 8 control values:

0, 0.25, 0.65, 0.85, 1.0, 1.0, 1.0, 1.0

Tools/Graph Profile -- Interactive (key-drag) editing

View / Display / Profiles / Graph has provided a way to visually relate to graphs and relabels. In SurfaceWorks 6.1 and beyond, this feature has been enriched to allow interactive editing of BGraph and Relabel control-point values.

To activate this, select a BGraph or Relabel object and invoke Graph Profile. The graph is displayed, as usual. You can click on any control point (the small circles), which gets marked with a red diamond, indicating it is the "active" control point. Then you can take these actions:

1) Holding down the <Ctrl> key, use the up or down cursor (arrow) keys to nudge the control point up or down, in coarse increments of 0.10.
2) Holding down <Ctrl> and <Shift> keys, use the up or down cursor (arrow) keys to nudge the control point u or down, in fine increments of 0.01.
3) Holding down the <Ctrl> key, use the right or left cursor (arrow) keys to move to the next or previous control point.

As you modify the graph control points, the model and any open views will dynamically update to reflect the changes.

Key dragging terminates when you release the <Ctrl> key, or move the focus to another window; then the red diamond disappears.

Grid

View>Grid or **G**

Toggles on / off the display of a lined grid in graphics display. Grid lines are visible only in orthographic views perpendicular to one or two of the coordinate axes — the six standard views (with perspective turned off) are among these. Heavy grid lines (every 10th line) are labeled and the Status Bar reports the size of the smaller grid divisions. E.g. Grid 0.1 means the light grid lines are 0.1 unit apart and the heavy grid lines would then be 1 unit apart (every 10th).

Grid is reset to off each time you open a model.

To display the lined grid:

Click or choose **View>Grid** or press **G**.

Help Menu

See "Help Topics" on page 178.
Licensing See "Licensing" on page 190.
See "About MultiSurfSurfaceWorks" on page 133.

Help Topics

Help>Help Topics

Opens a window that displays the Contents, Search, and Index tabs for SurfaceWorks' online Help.

To access SurfaceWorks Help:

Choose **Help>Help Topics**.

Hidden

See "Select Hidden" on page 211.

Hide

Show-Hide>

Hide Unselected

Hide Selection Set or **Ctrl+H**

Hide Points

Hide Parents

Hide Children

Hide All

Hide makes an entity or group of entities not visible. Entity(s) remain hidden until they are made to show again.

Here are some quick hide options that operate on specific groups of entities. **All** and **Points** ignore the contents of the Selection Set.

Unselected — hides all entities not currently in the Selection Set.
Selection Set — hides the entire Selection Set.
Points — hides all points in the model.
Parents — hides the first-generation parents of the selected entity(s).
Children — hides the first-generation children of the selected entity(s).
All — hides all entities in the model.

To hide the Selection Set:

1 **Select** the entities you want to hide.

2 Click or choose **Show-Hide>Hide Selection Set** or press **Ctrl+H**.

To hide All:

Simply click or **Show-Hide>Hide All**.

To hide all point entities:

Click or **Show-Hide>Hide Points.**

To hide Parents or Children:

1 **Select** the entity or entities whose parents or children you want to hide.

2 Click or **Show-Hide>Hide Parents**; or click **Show-Hide>Hide Children.**

To hide Unselected entities:

1 **Select** entities you want to remain visible.

2 Click or choose **Show-Hide>Hide Unselected.**

Icons

See "Arrange Icons" on page 135.

IGES

See "Export IGES" on page 168.

See "Import IGES" on page 181.

Import DXF

File>Import>DXF

Imports many forms of data from a 2-D or 3-D DXF file into an open SurfaceWorks model. DXF files conforming to AutoCAD™ R12, R13, R14 and 2000 are all handled.

The controls in the "Import DXF – Options" dialog allow you to control several aspects of the import operation and the conversion of data.

To import a DXF model into SurfaceWorks:

1 Choose **File>Import>DXF** and specify the .DXF file you want.

2 In the **Import DXF - Options** dialog under **AutoCAD Entities**, check the boxes for the **entity types** in the DXF file that you want translated and imported into SurfaceWorks.

3 **Choose the Insert Frame** you want to use for the geometry. If no Frame or Point entity is picked when you choose Import DXF, the default frame is used here.

4 **Choose a Point merging tolerance**. Default is 0.

5 **Choose a scale factor** for the geometry. Default is 1.

6 **Check** the **Points use Frame as Parent** check box if desired. Default is checked.

7 **Preserve AutoCAD colors** if desired. Default is unchecked.

8 **Choose layer** for geometry. Click **OK**.

Import DXF Options

AutoCAD Entities checkboxes let you choose which entities will be converted.

POINTs are converted to Points.

Points in DXF files are handled in two different ways depending on whether they have children.

- points that have no children are only imported when the POINTs box is checked
- points that are parents of other entities are only imported when that entity is imported

LINEs are converted to Lines;
ARCs and CIRCLEs are converted to various types of Arc;
POLYLINEs (including LWPOLYLINEs) are converted to type-1 BCurves;
SPLINEs are converted to NURBCurves.

Insert Frame. An Insert Frame can be preselected, or can be chosen by name from the dropdown. Any Frame or Point entity qualifies. Once chosen, the frame remains the default until another frame is chosen or the model is closed. (A Point serving as a frame means a parallel copy of the global coordinate system, translated so its origin is at the point.) The default frame '*' is the global coordinate system. When a frame other than '*' is chosen, the DXF data is interpreted as x, y, z coordinates in the insert frame. Thus, the imported objects are translated and rotated to place them in the orientation of the Insert Frame.

Point merging. Importing a DXF containing qualifying data usually generates some points. For example if you import a LINE, its two endpoints are created and connected with a SurfaceWorks Line. The "Point merging tolerance" edit box allows control over merging or substitution of existing points for ones that would otherwise be created. For example, if an ARC being converted has one of its ends at a location where another point already exists, within distance tolerance, the existing point will be used and creation of a new point at that location will be bypassed. If tolerance is negative, no merging is done, and every entity gets its full complement of new points. If tolerance is zero, points that have exactly the same location will be merged. A small positive tolerance – say on the order of .000001 times model size -- allows the incoming data to connect to existing data where possible, and generally produces a more "relational" model. Too large a tolerance risks loss of detail; for example, in the case of a LINE shorter than tolerance, the two endpoints will be merged, so the resulting SurfaceWorks Line will have zero length.

Scale factor. As noted under "units", under most conditions units conversion is automatically performed during import. This control is a provision for additional scaling of the imported geometry. For example, if you want to reduce the size of imported objects to 75% of their size in the DXF file, enter a scale factor of 0.75.

Points use Frame as Parent. As noted under "Point merging", import of DXF data usually generates points; by default these are Points with the Insert Frame as the

Frame parent so that the imported geometry acquires a durable dependence on the insert frame. If the frame is later moved or reoriented, the imported geometry will move along with it. This checkbox provides the alternative of generating Points with the default Frame as the Frame parent for all points instead. If the default Frame is used for the Insert Frame, this checkbox has no effect, but if another Frame is used for the Insert Frame, the geometry will be imported in the same location as if the points were using the Insert Frame as the Frame parent, but the actual Frame parent is the default frame. This would be equivalent to importing the geometry with the box checked, and then doing a group edit on all of the points and changing the Frame parent to the default entity.

Preserve ACAD Colors. This option controls the colors of imported objects. If "Preserve ACAD colors" is checked, objects will receive the SurfaceWorks color closest to the AutoCAD™ color specified in the DXF file. Otherwise, SurfaceWorks will use its default colors, e.g. cyan for arcs and splines, brown for lines, green for meshes.

Layer. A layer for the imported objects can be specified. DXF import ignores the layer settings in the DXF file, and all objects generated during an import are inserted on the same layer. This can be selected by name or number. The default layer is 0.

Units. Units are automatically converted. So for instance, if the DXF file is in millimeters and the SurfaceWorks model is in inches, the DXF entities will have their dimensions converted to the appropriate number of inches and the size of the imported geometry will not be affected. Conversions for files with no units is more complicated:

	DXF - no units	**DXF - Units**
MS2 - no units	No conversion	MS2 file Model Units is set to DXF unit and DXF entities are imported
MS2 - units	DXF entity's dimensions keep mag-nitude and assume units of MS2 file	DXF dimensions converted as described above

Import IGES

File>Import>IGES

Lets you import IGES geometry into SurfaceWorks. SurfaceWorks reads the file and automatically creates the specified surfaces and / or curves and their parent geometry. Most of the surface and curve types and all parent points are fully editable in SurfaceWorks (a few importable surface types are not directly editable; see "Correspondence Between IGES and SurfaceWorks entity types" on the next page.

Note: It is most types of surfaces and curves that are editable, not necessarily most actual surfaces and curves.

To import an IGES surface model into SurfaceWorks:

1 Choose **File>Import IGES** and **specify the .IGS file** you want.

2 In the IGES Import - Options dialog under **IGES Entity Types**, check the boxes for the **entity types** in the IGES file that **you want translated and imported into** SurfaceWorks.

3 If you choose **128 - Rational B-spline Surface**, the **Use Lightweight NURBS** check box under SurfaceWorks **Options** becomes available. Click this to turn on or off. Default is on (checked). For more on this,see "Use Lightweight NURBS" on page 183

4 Click **OK**.

There are reasons why you might want to specify import of only selected entity types:

- You only want curves, or surfaces.
- You only want B-spline surfaces.
- Since IGES files are often big and hence generate a lot of data when they get converted, you may want to only first keep certain types to see how it looks, before engaging into a time-consuming conversion.

However, since SurfaceWorks produces relational geometry, you will most certainly get parent entities of types you have not chosen for conversion. For instance, if you want trimmed surfaces to be converted and there is in the IGES file a trimmed surface, based on a B-spline surface and trimmed with composite curves, the B-spline surface and the composite curve will be converted by SurfaceWorks so that in SurfaceWorks, the trimmed surface appears as a child of the B-spline surface and the curves which are components of the composite curve. Oftentimes, additional intermediate geometry may have to be created to establish proper relationships between entities.

Note: In this instance the base surface will be located on a turned off layer and the parent curves will be hidden.

Correspondence Between IGES and SurfaceWorks/ Entity Types

The following table shows the correspondence between IGES entity types and SurfaceWorks entity types.

Note: This correspondence does not go both ways. For example, although an IGES 100 - Circular Arc will come into SurfaceWorks as an Arc, an Arc will export from SurfaceWorks as a NURBS curve.

IGES Entity	SurfaceWorks Entity
100 - Circular Arc	Arc
102 - Composite Curve	Will be exploded into its component curves.
104 - Conic Arc	An elliptical or a hyperbolic arc is translated as a Conic. A parabolic arc is translated as a degree-2 C-spline Curve.

106 - Copious Data	Only copious data of form number 1 (set of co-planar points), 2 (set of points), 3 (set of pairs (point, vector); the vector part is ignored), 11 (planar piecewise linear curve, also called polyline or line string), 12 (piecewise linear curve), 13 (piecewise linear curve with vectors associated with each vertex; the vectors are ignored) and 63 (simple, meaning non-self intersecting, closed planar curve) are supported. The piecewise linear curve is translated as a degree-1 B-spline Curve.
110 - Line	Line
112 - Parametric Spline Curve	B-spline Curve
116 - Point	Point
126 - Rational B-Spline Curve	NURBS Curve
130 - Offset Curve	Relative Curve
141 - Boundary	snakes
142 - Curve On Parametric Surface	snakes
108 - Plane	2-point Plane
114 - Parametric Spline Surface	
118 - Ruled Surface	Ruled Surface
120 - Surface Of Revolution	Revolution Surface
122 - Tabulated Cylinder	Translation Surface
128 - Rational B-Spline Surface	NURBS Surface
140 - Offset Surface	SurfaceWorks Offset Surface
143 - Bounded Surface	Trimmed Surface
144 - Trimmed Surface	Trimmed Surface

Note: Although there is no Edit dialog for some entities that can be translated and imported into SurfaceWorks, these entities are relational entities with parents and you can change these entities in a more indirect manner. To change the shape, move the parent points. To change properties such as layer, color, visibility, or divisions, do a multiple-entity edit — that is, select the entity twice (use Ctrl+shift+click for the second select) or select two or more entities that share the property you want to change, then Edit Definition. Properties such as type or degree are not editable.

Use Lightweight NURBS

When SurfaceWorks reads in an IGES entity type 128 - Rational B-spline Surface, it converts it to a single NURBS Surface entity, supported by, among other things, the number of (say N) control points that make up the original NURBS surface. If N is large, which often happens, the number of entities that are created in SurfaceWorks bogs down the system, resulting in slow performance and response in a variety of operations. When "**Use Lightweight NURBS**" is checked, these control points are not each created in SurfaceWorks, but instead become part of the Light NURBS Surface that is created. The data is not lost, just part of a larger entity that includes that data within itself. Unlike with a "heavy" NURBS Surface, there is no access to any of the control points, but it is hard to imagine doing anything useful with the individual control points anyway, when there are so many of them.

Layer Conventions

SurfaceWorks uses the following layer conventions when importing IGES geometry.

- The imported IGES surfaces will be put on the layer (level) they were assigned in the IGES file.
- Any construction entities SurfaceWorks created when importing IGES surfaces (parent points and / or curves) will be put on Layer 253 (this layer is initially turned off and these entities, therefore, are neither visible nor selectable until Layer 253 is turned on.) These are the entities you can edit to refine or otherwise modify the imported NURBS Surfaces.
- “Consumed” entities – for instance, the parent surface of a surface produced by trimming, are on Layer 254, also turned off initially.
- If any imported entities cannot be properly evaluated, they will be in error and will be put on Layer 252. These entities are not visible because they are in error.
- The parents of entities in error will be put on Layer 251. These are your graphic handles leading to the entities in error.

Import .SLDCRV

File>Import>.SLDCRV

Imports a SolidWorks .SLDCRV file of 3D data points and creates a SurfaceWorks curve from it. An imported curve and its control points can be used as a parent of other geometry, just like any other SurfaceWorks entities. For instructions about how to make a .SLDCRV file, see ”SLDCRV” on page 218” .

To import .SLDCRV data and fit it in SurfaceWorks:

1 Choose **File>Import .SLDCRV** and **specify the .SLDCRV file** you want.

2 The import dialog will report the number of data points in the input file.

3 **Click the radio button selecting the method of curve construction** you would like to use with the data points, **or to import the points only**. Note: if you specify Points Only, B-spline Curve, or C-spline Curve, the Number of Control Points option will gray out (each of these options uses the full set of points).

4 **Fill in the rest of the data** (or accept the defaults), then click **OK**.

5. **Check the results**. If they are satisfactory, continue modeling. If they are not satisfactory, Undo and try specifying a different number of control points (if applicable), a different degree (if applicable), or a different curve fit option.

SLDCRV Import Options

Points Only. No curve is created; each data point in the file results in creation of one Point entity.

B-spline Curve. The points read from the file will become Point entities and will be used as the control points of a B-spline Curve of the specified degree.

C-spline Curve. The points read from the file will become Point entities and will be used as the control points of a C-spline Curve of the specified degree. The resulting curve will pass through all the data points in sequence.

Curve Fit: There are two options. In either case, the data points read from the file are least-squares fitted with a B-spline curve, using the specified number of control points. The control points become Point entities. Generally, the control points are fewer in number than the data points, and in different locations, but the first and last control points will always be located at the first and last data points. Two alternative curve fitting algorithms are provided, adapted to different qualities of data:

Smooth Input Data. This procedure is best adapted to data points having smoothly varying spacing, such as might arise from carefully spaced sampling or generation from analytic formulas.

Irregular Input Data. This procedure is designed to accept completely irregular data spacing. It will function with smoothly spaced data as well, but may not achieve as close a fit with a given number of control points.

Name. Accept the default offered or type in your own. SurfaceWorks guarantees that defaults offered will be unique. If you type in a new name, SurfaceWorks will check it for name conflict and if needed will generate a unique name.

Point entities are given the curve name with pt1, 2, 3... appended.

Degree. Choose degree from the dropdown list: 1 = linear (needs at least two control points), 2 = quadratic (needs at least three control points), 3 = cubic (needs at least four control points), etc. For B-spline curves, including fitting options, degree options are 1 - 5; for C-splines, degree options are 1 - 3.

Number of control points. Only for Fitted Curves (B-splines and C-splines use the input data control points). Specify the number of control points SurfaceWorks is to use to fit the input data.

Number of control points tradeoff: The fewer the control points, the smoother the resulting curve, but (in general) the RMS deviation will increase.

Reverse order of points. Check this box if you want the SurfaceWorks curve to use the input data points in the order opposite to their listing in the input file.

To See the Data Points and the Fitted Curve

To see the data points as well as the fitted curve and its control points:

1. Import the .SLDCRV file as **Points Only**.

2 Select all of the points, **Edit Definition** and **change their color**. You might well **move them to a vacant layer** at the same time, so they're easy to get rid of when you're done with them.

3 Import the same file using a **Fit option**.

Creating a .SLDCRV File

To create a .SLDCRV file in a text editor or another program:

A .SLDCRV file is an ASCII text file containing X, Y, Z values in three columns. Each line represents one 3D point, and must contain three decimal values separated by spaces or tabs. The file may contain units designators (in, mm, ft, m, c) following the decimal value, with no space between. If the file contains no units, the units of the SurfaceWorks part will be assumed. If the file units are different from those of the SurfaceWorks part, a units conversion will be made during import.

The following is an example of a complete .SLDCRV file containing five points:

```
3.000in  2.500in  2.333in
4.000in  3.250in  2.333in
5.000in  3.500in  2.333in
6.000in  3.250in  2.333in
7.000in  2.500in  2.333in
```

Import Triangle mesh

File>Import>Triangle mesh

Imports an STL or RAW file of triangular meshes and creates a SurfaceWorks TriMesh from it.

An STL (STereo Lithography) file consists of a list of facet data. Each facet is uniquely identified by a unit normal (a line perpendicular to the triangle and with a length of 1.0) and by three vertices (corners). The normal and vertices are each specified by three coordinates, so there is a total of 12 numbers stored for each facet.

A RAW file also consists of a list of facet data. As ouput by RHINO software, each facet is uniquely identified by three vertices (corners). The vertices are each specified by three coordinates, so there is a total of 9 numbers stored for each facet.

Insert

Insert>

Point	**Curve**	**Lofted Surface**	**Solid**	**Contours**
Bead	**Snake**	**Surface**	**Frame**	**Entity List**
Magnet		**Composite Surface**	**Plane**	**Relabel**
Ring		**Triangle Mesh**	**Wireframe**	**Graph**
Knotlist	**Variable**	**Formula**	**Text Label**	**Copies**

The **Insert** menu contains options for creating new entities. (For details on **Copy Last**, **Quick Points Mode** and **Quick Spline Mode**, please refer to their descriptions elsewhere in this chapter.)

To create an entity:

1. **Select** any parent entities required by the new entity, paying attention to order of selection. (This step is optional.)

2. Choose the class of entity you want to create from the **Insert** menu, or use the toolbar icons provided for certain entities and entity classes, as follows:

   - Points - Choose **Insert>Point** or click one of the following toolbar buttons:

      Point  Bead

      Magnet  Ring

   - Curves - Choose **Insert>Curve** or click .

   - Snakes - Choose **Insert>Snake** or click .

   - Surfaces - Choose **Insert>Surface** or click one of the following toolbar buttons:

      Lofted Surface (button with cyan control curves)

      Surface (button with all blue boundary)

      Composite Surface

3. Choose the specific type of entity you want to create from either the class or toolbar submenu. The Insert dialog for the entity is displayed.

4. Change whichever default properties and parents you want. If you did not preselect parents above, you must specify them now to replace *EMPTY*.

5. Click **OK**.

The newly created entity is automatically selected, replacing previous contents of the Selection Set. To reverse the creation of an entity, you can simply delete it (it will have no children, if it was the last entity you created); or you can use **Edit>Undo**.

For details on creating entities in general, see " Creating Entities". For details about the parents and characteristic properties required to define specific entities, see " Entity Descriptions."

Insert Menu

See ""Quick Point Mode" on page 202".
See """Quick Spline Mode" on page 203".
Digitizer Mode See "Using the Digitizer"
See ""Copy Last" on page 155".

See ""Copies" on page 153"
Entities See ""Insert" on page 186".

Invert Selection Set

Select>Invert Selection Set

Puts into the Selection Set all entities not currently in the Selection Set, regardless of filters, and removes all entities that were in it.

General usage:

1. **Select** the entities you don't want in the Selection Set. (Presumably this is a smaller number of entities than those you eventually want in the Selection Set, otherwise you'd just select the ones you do want!)
2. Choose **Select>Invert Selection Set**.

Last

See ""Copy Last" on page 155".

Layers

Tools>Layers or **L**

Lets you set the current layer, turn layers on/off, and name layers. When a layer is off, all the entities on that layer are off, out of the picture (although they still exist in the model) — none of the entities on the turned-off layer is shown in the graphics window and none of them is available for selection in the Available Entities pane.

SurfaceWorks supports layers 0-255. In SurfaceWorks, the startup default for current layer is Layer 1. The layers can be turned on/off in any combination, and they can be named.

An entity can be on only one layer. You specify the layer of an entity in its properties (see "Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Layer").

Layer on/off status and Layer names are saved with the model.

To set layer options:

1. Choose **Tools>Layers** or **L**.
2. Set the layer options to suit your needs.
3. Click **OK** to accept the changes or click **Cancel** to discard the changes.

Current Layer. Specify from 0-255. The current layer is the default layer for any entities created. The Current Layer's name is displayed alongside its number, at the top of the dialog. The "Current" button. You can choose a new layer for the current layer by highlighting its number/ name in the list box and clicking the "Current" button. The number of the current layer is displayed on the right side of the Status Bar. It appears as L: followed by the current layer number, e.g. L:1.

You also can constrain selection to just the current layer, using **Select>Current Layer Filter**.

Layers with entities on them. Layers that have one or more entities on them are marked with an asterisk (*).

Turning layers on and off. Check boxes toggle the layers on (checked) or off (empty). You need to click the check box itself (not just anywhere on the line in the list). You also can change the on / off layer settings using the following buttons:

Set All turns all layers to on.

Clear All turns all layers to off.

Invert reverses the on / off state of each of the layers, turning off all the layers that were on, and turning on all the layers that were off.

Move Up

Move Down To insert one or more layers, use the "Move Down" button would be used. You can open space for a new layer by highlighting a layer (say, layer 5) and clicking the “Move Down” button. The contents and name of layer 5 are moved down one layer; if layer 6 is already occupied, its contents and name are moved to layer 7 etc, until we reach an empty layer, where the chain reaction stops. Thus, an empty layer is created at layer 5. If no insert can be done, the Insert Layer button is deactivated.

With a layer selected, and the cursor held over the "Move Down" button, arrows appear in the layer list indicating how many layers will be affected by the move. Multiple layers can be selected to be moved. In this way a block which may contain blank layer separators, can be moved as a group.

The "Move Up" button operates in the same manner as "Move Down", but in the opposite direction.

Move Layer “Move Layer…” button. You can move the contents and / or name of any selected, non-empty, layer to another selected layer. This button opens a dialog where you set check boxes that determine whether you move the layer name and the layer on / off state with the layer contents. When a layer is empty and cannot, therefore, be moved the Move Layer button is deactivated.

Swap The contents and names of two selected layers can be swapped with each other.

Naming layers. Click the layer you want to name. That layer will become highlighted and its number will appear in the small gray box to the left of the name field. Click the cursor in the name field below the list box, type in the layer name you want, and then click the **Rename** button.

Layer Settings dialog and Undo

As long as you stay in the Layer Settings dialog, individual “moves” and “inserts” are not recognized as undoable events. If you leave the dialog by the “Cancel“ button, all changes made while in the dialog are undone, and the model is exactly the same as when you entered the dialog. Leaving the dialog by accepting (clicking OK) registers as one undoable event, covering all the changes made.

Licensing

Help>Licensing

Opens the SurfaceWorks License dialog box.

To access SurfaceWorks Licensing:

Choose **Help>Licensing**.

Macros

Tools>Macros

See "Macros Dialog" in the "Macros" chapter.

Mass Properties

Tools>Mass Properties

[Available in SurfaceWorks standalone, and in SolidWorks Integrated mode with SurfaceWorks Marine]

Opens a read-only text window and tabulates:

- center of gravity (location) for points
- arc lengths and centers of gravity for curve, snake, and contours entities
- areas and centers of gravity for surface entities
- the 6 moments of inertia of point, curve, snake, contours, and surface entities

Entities that are NOT INCLUDED in the Mass Properties calculations:

symmetry image entities
hidden entities
entities on a layer that is turned off

If you want to keep the results as a disk file (which you could also print if you so chose), use the **Right-mouse-click** menu and choose **Save As Text**.

The moments of inertia are purely geometric quantities, with units of $length^3$ for curves and $length^4$ for surfaces. To convert them to actual mass moments of inertia, you multiply by a density (mass/ unit length for curves, mass/ unit area for surfaces). Because the moments of inertia can have a very large range of values, they are displayed in exponential format (also called scientific notation) — for example, 3.45e4 means 3.45×10^4 or 34,500.

All moments of inertia are with respect to the X, Y, Z world coordinates. To convert to moments of inertia about another point, such as the model center of gravity, you would need to use the parallel axis theorem. (Ref. J. L. Meriam, Dynamics, 2nd Ed., Wiley, New York, 1971.)

Sample output (using DEMO.MS2):

```
Mass Properties     27-Jul-2001  14:45:09
Model file: DEMO.MS2

Points
  Entity     Object Name              X       Y       Z
```

```
  FramePoint  P11           1.00    0.000   0.000   3.600
Curves
  Entity     Object Name   Length     X       Y       Z
  BCurve      MC1           5.58    1.272   0.000   1.133
  BCurve      MC2           7.32   15.000   3.192  -0.056
  BCurve      MC3           5.26   30.000   2.283   0.851
Contours
  Entity     Object Name   Length     X       Y       Z
  Contours    stations     68.86   16.236   2.680   0.312
  Contours    waterlines   29.92   15.930   3.122   0.000
Surfaces
  Entity     Object Name    Area      X       Y       Z
  CLoftSurf   hull         189.00  15.943   2.675   0.323

Moments of inertia (about origin)
Points   (ft^2)
  Object Name   IXX       IYY       IZZ       IYZ       IZX
IXY
  P11      1.30E+01  1.30E+01  0.00E+00  0.00E+00  0.00E+00
0.00E+00
Curves   (ft^3)
  Object Name   IXX       IYY       IZZ       IYZ       IZX
IXY
  MC1      1.77E+01  3.04E+01  1.27E+01  0.00E+00  1.94E+00
0.00E+00
Contours   (ft^3)
  Object Name   IXX       IYY       IZZ       IYZ       IZX
IXY
  stations 7.53E+02  2.26E+04  2.32E+04  1.17E+02  2.90E+02
3.23E+03
Surfaces   (ft^4)
  Object Name   IXX       IYY       IZZ       IYZ       IZX
IXY
  hull     2.08E+03  6.02E+04  6.17E+04  3.24E+02  7.69E+02
8.73E+03
```

Printing

Whenever Mass Properties is open, the context sensitive menu (right-click) will provide access to a Page Set-up and Print menu choice.

Measure

See “ ”Angle” on page 134”.

See “ ”Clearance” on page 138”.

See “ ”Distance” on page 159”

Mechanical View Orientation

View>Mechanical View Orientation>

 **Front**  **Left**  **Top**

 **Back**  **Right**  **Bottom**

Home

Lets you set the view orientation to one of the seven standard views: Front, Back, Left, Right, Top, Bottom or Home.

If you are in Perspective view, using any of these buttons causes SurfaceWorks to go to Orthographic view.

Resets to Home view each time the model is opened.

To change the orientation via the menus:

Choose **View>Mechanical View Orientation** and then **Front, Back, Left, Right, Top, Bottom** or **Home**

To change view orientation via the toolbar buttons:

Click the appropriate toolbar button:

Mode

See “”Quick Point Mode” on page 202”

See “”Quick Spline Mode” on page 203”

See “”Select Mode” on page 212”

Model Options

Tools>Options, Model

In Model Options, you can choose a plane or axis for symmetry, set the Home view and write comments that are saved with the file.

All of these option settings are saved with the model.

To choose symmetry for your model, set home view or write a comment for your file:

1. Use the **Tools>Options, Model** tab.
2. Make whatever changes you need to Symmetry and Home View settings.
3. Type a Model Comment if desired.
4. Click **OK**.

Symmetry

Mirror Symmetry. Set X, Y, and / or Z mirror symmetry images (check boxes). Any combinations of mirror symmetry are allowed.

Mirror symmetry is symmetry across one or more of the coordinate-axis planes. For example, most boat designs have complete (model-level) symmetry with respect to the Y=0 plane. In this case, it is generally advantageous to use Y-symmetry so you only define entities on one side of the Y=0 plane and the mirror image entities are implied by mirror symmetry. Or, suppose you design a shampoo bottle that is symmetric with respect to both the X=0 and Y=0 planes. In this case, it could be advantageous to use both X and Y symmetry; then the “real” entities would comprise only one quarter of the model.

Rotational Symmetry. Set X, Y, Z, or no (none) rotational symmetry images (radio buttons) and the number of images you want displayed — be sure to include the original copy in this number. (If you build one quarter of the model, you will need 4 copies to show a complete model). The program default for rotational symmetry is “None.” Which rotational options are available depends on mirror image settings, if any (if mirror symmetry is specified, the only rotational symmetry allowed is about the same axis as the mirror symmetry axis).

Rotational symmetry is symmetry around one of the coordinate axes. In rotational symmetry, n is the total number of copies to be displayed, including the original. n-fold rotational symmetry is present if the model looks the same after being rotated 1/n revolution (360/n degrees) about the indicated axis. For example, a 5-bladed propeller would have 5-fold symmetry about the shaft axis. In this case, you could design only one blade (and 1/5 of the shaft, hub, etc.); turning on the symmetry images would display the other 4 blades, which would be completely identical to the original (look at model file PROP3.MS2).

Mirror/rotational combinations

Rotational symmetry can be combined with mirror symmetry, but only to this extent: a model can have mirror symmetry with respect to the same axis as its rotational axis. For example, if rotational symmetry with respect to the Z-axis is specified, this can be combined with mirror symmetry with respect to Z.

If you do not specify symmetry, the model will have no symmetry images.

Model Symmetry

If your model has mirror symmetry across one or more of the coordinate axes, or rotational symmetry around one of the coordinate axes, you can make use of SurfaceWorks' symmetry images by creating points, curves, and surfaces that represent a portion of the model (we call this the “real” part of the model), then turn on the symmetry images to see the entire model.

View>Symmetry Images (or <F5>) is a toggle, allowing you to display the model with all images showing or with only the “real” entities visible.

Note 1: If your model has only partial symmetry, or symmetry about an axis other than one of the coordinate axes, use mirror and rotated entities (Mirrored Surfaces, etc.) rather than symmetry images.

Note 2: If you will be using the weight schedule calculations, symmetry images may affect the weight values you assign to some entities. See “Reference Guide to Functions - Weight Schedule".

Home View and Model Comment

Home View

“Home” view is the view in which SurfaceWorks first displays a model on the screen. It is also the view you can return to at any time by pressing  or the <Home> key. The Home view can be orthographic (default) or perspective. If you turn off the orthographic checkbox, the view will be perspective rather than orthographic. If you do not specify Latitude, Longitude for this view, the program will use the default Home view: Latitude -30, Longitude 60, orthographic.

Model Comment:

This is a place to record any comments that will be saved with your model.

Mouse Wheel Zoom Preferences

Click the check box to reverse the mouse wheel zoom directions.

Model Units:

In Model Units, you can set or change the length unit used for the model and if changing the units either scale to the new unit or just change the designator.

This option is saved with the model.

To set or adjust length units:

1 Use the **Tools>Options>Model** tab.

2 Make whatever changes you need to the length unit settings.

3 Click **OK**.

Length unit:

When starting a new model choose the length unit for it here by clicking the appropriate radio button. If your model already exists, clicking here will change the length unit of all of the entities.

Convert to new units. If you change units for a model that already has geometry, the size of the geometry remains unchanged, it is just measured in the new units. In this case, a one meter segment would become 3.28 feet. If you want to scale the geometry, so that entities that are one meter become one foot, for instance. First change to the new units, and then scale the model using Edit>Transform>Scale. You will have to calculate the appropriate factor so that the model ends up the size you want.

Mouse Wheel Zoom Preference

The setting, to reverse the mouse wheel zoom direction, can be found on the General tab of the Options dialog.

Nametags

View>Nametags>

	**Point**		**Surface**		**Triangular Mesh**	**Contour**
	**Curve**		**Composite Surface**		**Plane**	**Frame**

Lets you toggle the display of entity nametags in graphics displays. The switches are by entity class (Point, Curve, Surface, Plane, Contours, Frame); they can be on in any combination. When Nametags are on, SurfaceWorks displays the names of all entities of the specified class(es) currently visible on the screen.

All Nametags reset to off each time a model is opened.

To toggle nametag display via the menus:

1 Choose **View>Nametags**.

2 In the Nametags dialog:

check the boxes of the classes for which you want to display nametags
and / or
uncheck the boxes of the classes for which you don't want to display nametags

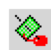
3 Click **OK**.

Make Default. Click this button to make the current settings the program default. Next time you invoke SurfaceWorks, nametags checked in the default set will automatically be turned on.

To toggle nametag display via the toolbar buttons:

Click the appropriate toolbar button; in (light gray background) = ON, out (darker gray background) = OFF:

 Point  Surface  Triangular Mesh

 Curve  Composite Surface  Plane

Nametag Display

Entity nametags are displayed in the entity's color. They are located by the following rules:

- **point**: the name is positioned just to the lower right of the point
- **curve**, **snake**: the name is positioned next to the midpoint of the entity
- **surface, composite surface, triangular mesh**: the name is positioned at the average location of entity points plotted on the screen
- **plane**: the name is positioned off one of the "corners" of the plane representation
- **contours**: the name is positioned approximately at the middle of the middle contour
- **frame**: the name is positioned just to the lower right of the frame's origin

Sometimes, Nametags are pasted partially or completely over other Nametags, so some names seem to be shortened or missing. This usually can be clarified by zooming in or moving the viewpoint, or checking the Transparent nametags option on the **Tools>Options, General** tab.

New

See "New File" on page 196.

See “”New Model Window” on page 196”

New File

File>New or **Ctrl+N**

[Not available in SolidWorks Integration mode.]

Creates a new, empty SurfaceWorks model.

Choose **File>New**.

New Model Window

Window>New Model Window

To open a new model window:

Choose **Window>New Model Window**.

Open

File>Open or **Ctrl+O**

[Not available in SolidWorks Integration mode.]

Opens an existing SurfaceWorks model.

Choose **File>Open**.

Open Windows List

Window

Allows you to activate any of the windows open in SurfaceWorks. The open windows are listed at the bottom of the Window menu in the order they were opened. The active window has a checkmark.

To activate a different window:

1. Choose **Window**.
2. **Click** its name in the list

Options

See “Display Options” on page 158”.

See “Dragging Options” on page 161”.

See “Entity Options” on page 163”

See “General Options” on page 175”.

See “Model Options” on page 192”.

See “Performance Options” on page 198.

See "Session Options" on page 213

See ”Surface Curvature Options” on page 220".

See “”Tools Options” on page 224”.

Orientation

See "Auto Orientation" on page 135.

See "Entity Orientation" on page 164.

See “Mechanical View Orientation” on page 191.

Orthogonal Dragging

See “Orthogonal Dragging” on page 197

Pan

View>Modify>Pan or **Ctrl+Arrow keys** or **Ctrl+center mouse wheel / button**

Allows you to use the mouse to Pan the view. This has the effect of changing the "look point" of the imaginary camera through which the model image is viewed. The cursor changes to while in Pan mode.

To use the center mouse wheel or button, hold down the **Ctrl** key while pressing the center wheel or button and moving the mouse.

To pan the view of the model:

1. Click , or choose **View>Modify>Pan**, or **right-click in graphics whitespace** and choose **Pan** from the shortcut menu to get into Pan mode, and then hold down the left mouse button or hold down the **Ctrl** key and then **press the center mouse wheel** or **button**.

2. Drag the mouse until you have the view you want.

3. Drag again, as desired (as many times as you want).

Note: You can also Pan by preset increments using the keyboard Ctrl+Arrow keys.

To get out of Pan mode you can do any of the following:

- Click or choose **View>Modify>Pan** again.
- Press the **Esc** (Escape) key.
- **Right-click in graphics whitespace** and choose another cursor mode.
- Choose **another cursor mode** (e.g. Rotate, Zoom In / Out, or Select mode) via the toolbar buttons, the right-click shortcut menu, or the View menu.

Parent

See "Select Parents" on page 212.

See "To hide Parents or Children:" on page 179.

See "To show Parents or Children:" on page 217.

Parent/Child

Right-click an entity, Parent/Child

Many SurfaceWorks entities depend on one or more previously-created entities for part of their definition. These dependent entities are "child" entities. For example:

- A Point with a non-default Point Parent is a child of the point entity from which its coordinates are offset.
- A lofted surface is a child of the control curves that shape it.
- A SubSurface is a child of the surface on which it is built.

An entity upon which a child entity is built is a "parent" entity.

To view parent/child relationships:

1. **Right-click an entity** in the graphics display, in the Surface Manager, or in the Selection Set or Available Entities panes.
2. From the shortcut menu, select **Parent/Child**. You can only view the relationships here. To edit a relationship, you'd need to edit the definition of the entity itself.
3. Click on any of the entities in the Parent/Child dialog to see its parents/children.

Performance Options

Tools>Options>Performance

Lets you set:

- Screen repaint rates for the shaded and wireframe displays
- The number of degrees per keystroke for rotating the model using the arrow keys
- The number of decimal places for linear coordinates
- Divisions multiplier

  The decimal places value and the Divisions multiplier are saved with the model; the other values are saved with the application.

Lets you reset the bounding box SurfaceWorks uses to display the model (Update Extents).

To specify performance settings:

1 Choose **Tools>Options>Performance**.

2 Specify your settings.

3 Click **OK**.

<u>Display Controls</u>:

Repaint shaded every _____ entities.
Repaint wireframe every _____ entities.

When the program draws the initial shaded or wireframe display, or redraws (e.g. after a viewpoint change via the arrow keys, an edit via an edit dialog, or clearing the Selection Set), you can have it:

- Draw the picture <u>all at once</u> after it has finished updating the entire mathematical model; Repaint value = 0 (Recommended setting).
- Draw the picture <u>in batches</u> of one or several entities, repainting the picture each time it has finished calculating the specified number of entities; Repaint value = a positive integer.

Note: After dynamic viewpoint changes (e.g. rotate, pan, and zoom via the menu or the toolbar buttons) and edits via dragging, redraw always is all at once.

The benefit of drawing all at once, especially on slower machines, is that it is faster overall. It may seem slower – because you don't get immediate feedback. So what you choose for these settings may be a tradeoff between speed and feedback. If your model has lots of entities (for instance, if you've imported an IGES file), you may want to redraw in batches, even on a fast machine.

For both shaded and wireframe display repainting, the initial program default is 0 which is the recommended setting.

Degrees per keystroke. This setting determines how far you rotate around the model with one press of an **arrow key**, and how far in/out you zoom with **PgUp** and **PgDn**. If you hold down **Shift** while pressing an **arrow key**, **PgUp**, or **PgDn**, the effect on viewing parameters is 1/10 that of the unshifted key. For example, if you are rotating and Degrees per keystroke is set to 10 degrees, the view orientation will change in 1-degree increments. The program default is 10 degrees per keystroke.

<u>Model</u>:

Decimal places. Set the number of decimal places, up to 6, displayed for linear measures. The program default is 3 places.

Divisions multiplier. The factor (1-10) that SurfaceWorks uses to calculate the subdivisions of all curves, snakes and surfaces. SurfaceWorks multiplies the number of each entity's subdivisions by this factor when it calculates subdivisions.

When the divisions multiplier = 1 (the default), the number of subdivisions remains unchanged. Setting the divisions multiplier to 2 - 10, allows you to increase the number of subdivisions by automatically multiplying all subdivisions by the factor.

This tool allows you to work with low divisions for speed, then to crank the divisions up uniformly when you need more accuracy, say for producing a construction drawing or exporting an IGES file.

Update Extents:

Click this button if the bounding box the program uses to display the model no longer centers and fills the window with the image when you Zoom to Fit.

Perspective

View>Display>Perspective or **P**

Displays the model in a perspective view. Perspective views are realistic 3D views very similar to those you would see with your eye or with a camera. In perspective, points in 3D are projected along lines that converge at the camera (or eye) point, and entities farther away appear smaller than those that are closer.

In orthographic views, on the other hand, the points are projected along parallel lines (usually parallel to one of the three coordinate axes).

SurfaceWorks provides perspective and orthographic views from any direction.

To display the model in perspective view:

Click  to its on (pushed in; light gray background) position or choose **View>Display>Perspective** or **P** (you only need to click the Perspective option if it is not preceded by a checkmark).

Note: This option actually is a toggle. When the menu item is checked, the display is perspective; when the menu item is not checked, the display is orthographic.

Point Dragging and Nudging

The Point entity has always had a dragging constraint or type, 0 to 7, but it has not previously had any significance. In SurfaceWorks 6.0, the type now has an important role: it controls constraints applied to the point during dragging and nudging. Only when a point uses Cartesian coordinates is the dragging constraint available. If a point employs Polar coordinates, the dragging constraint is not available. The 8 FramePoint types are as follows:

Type	Dragging / nudging behavior
0	Free to drag in any direction
1	x is pinned; the point is draggable in y and z
2	y is pinned; the point is draggable in x and z
3	x and y are pinned; the point is draggable only in z
4	z is pinned; the point is draggable in x and y
5	x and z are pinned; the point is draggable only in y
6	y and z are pinned; the point is draggable only in x
7	x, y and z are pinned; the point is not draggable

The order of the types in the above table may look a bit odd, but the logic is simple, and easy to remember with the following recipe:

As an example application of this feature, consider a boat hull design consisting of a lofted surface supported by a number of master curves in transverse planes. If you use FramePoints of type-1 (x pinned) to support these transverse master curves (in a frame parallel to the global coordinates, e.g.'*'), these control points can be dragged or nudged in any oblique view, and will remain in their respective transverse planes. If the forward master curve lies in the centerplane, you could use type-2 FramePoints (y pinned) to allow dragging in any view without moving it out of the centerplane. The last (centerplane) control point on each transverse master curve could be type-3 (x and y pinned), to keep it both in the centerplane, and in its correct transverse plane.

Point

See “Quick Point Mode” on page 202.

Previous View

View>Modify>Previous View or **F8**

The previous view is the most recent view (in the same window) that was allowed to redraw completely. Using **View>Modify>Previous View** successively toggles back and forth between two views.

To display the previous view:

Click  or choose **View>Modify>Previous View** or press **F8**.

Query/Breaks

Identifies any breaklines in a selected surface, or breakpoints in a selected curve or snake.

Query/Evaluation Times

Lists evaluation times for all entities that register 1 millisecond or more, sorted into descending order. This is a valuable troubleshooting tool for identifying and analyzing problems of slow performance.

Query/Extents

Identifies the geometric extents of the item(s) in the selection set or, if nothing is selected, the extents of the entire model. In the case of displaying the extents for the whole model, the entities which lie at the limits of the extents on each face of the bounding box will be listed.

Query/Host/Guest

Host - Identifies the host of the selected object.   Displays a dialog box that identifies the host(s) of the selected entity.

Guests - Identifies all guests (residents) of the selected entity.

You can click on an entity in either window to select it, and see its hosts and guests.

Query/Parent/Child

Parent - Identifies the parent of the selected object. Displays a dialog box that identifies the parent(s) of the selected entity.

Child - Identifies all children of the selected entity.

You can click on an entity in either window to select it, and see its parents and children

Query/Trimesh Properties

Query / Trimesh Properties reveals element counts and basic topological properties of a triangle mesh, or a parametric surface or Composite Surface that can be used as a triangle mesh: The information includes counts of nodes, links (edges), triangles, boundary links and boundary loops, and the topological genus. This example is for an octahedron:

TriMesh 'tm'

6 nodes

12 links

8 triangles

0 boundary links

0 boundary loops

Genus X 2 = 0

(0 = sphere, -1 = disc, -2 = disc w / 1 hole, etc.)

Quick Point Mode

Insert>Quick Point Mode, or **Q**

Allows you to create points by simply clicking on the screen in graphics display. The kind of point created, (Point, Bead, Magnet, Ring) depends on what you have preselected when you turn on Quick Point Mode. All the points you make (as well as any preselected point entities) accumulate in the Selection Set, allowing you to immediately continue on and insert a curve (or snake) parented by those points — so Quick Point Mode also is a short cut for curve (and snake) creation.

The quick point drawing option is a mode toggle — it changes the cursor into Quick Point Mode, until you select the option again, change to another mode (e.g. Select Mode) or hit the <Esc> key.

To turn Quick Point mode on:

1 **Preselect** an appropriate entity for the kind of points you wish to create:

- If you want to make Points in space, preselect nothing or any kind of point entity (to set the depth, if you so desire).
- If you want to make Beads, preselect a curve in space.
- If you want to make Magnets, preselect a surface.
- If you want to make Rings, preselect a snake.

2 Click  or choose **Insert>Quick Point Mode**, or press **Q**.

For further details on using quick points, see "Creating Entities - Quick Points."

Quick Spline Mode

Insert>Quick Spline Mode or **S**

[Set defaults using **Tools>Options>Entity** first.]

Allows you to create B-splines and C-splines (curves or snakes) by dragging out spline segments on the screen in the graphics display. An alternate spline creation method lets you simply click control point locations.

You set defaults for spline type, spline Degree, and spline creation mode on the Tools>Options>Entity tab. Whether curves in space or snakes are created depends on the kind of entity you have preselected when you turn on Quick Spline Mode. These settings are saved with the application.

The quick spline drawing option is a mode toggle — it changes the cursor into Quick Spline Mode, until you select the option again or change to another mode (e.g. Select Mode).

To set default options for Quick Spline Mode:

Use the **Tools>Options>Entity** tab. For details, see “ ””Tools Options” on page 224”

See "Creating Entities - Quick Splines".

Real Values

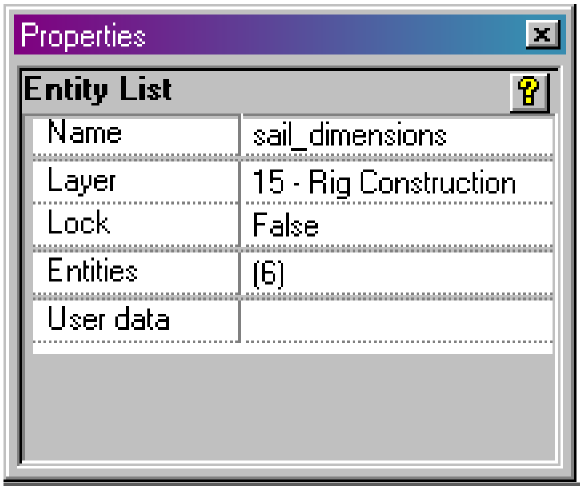
Tools>Options>Real Values or V

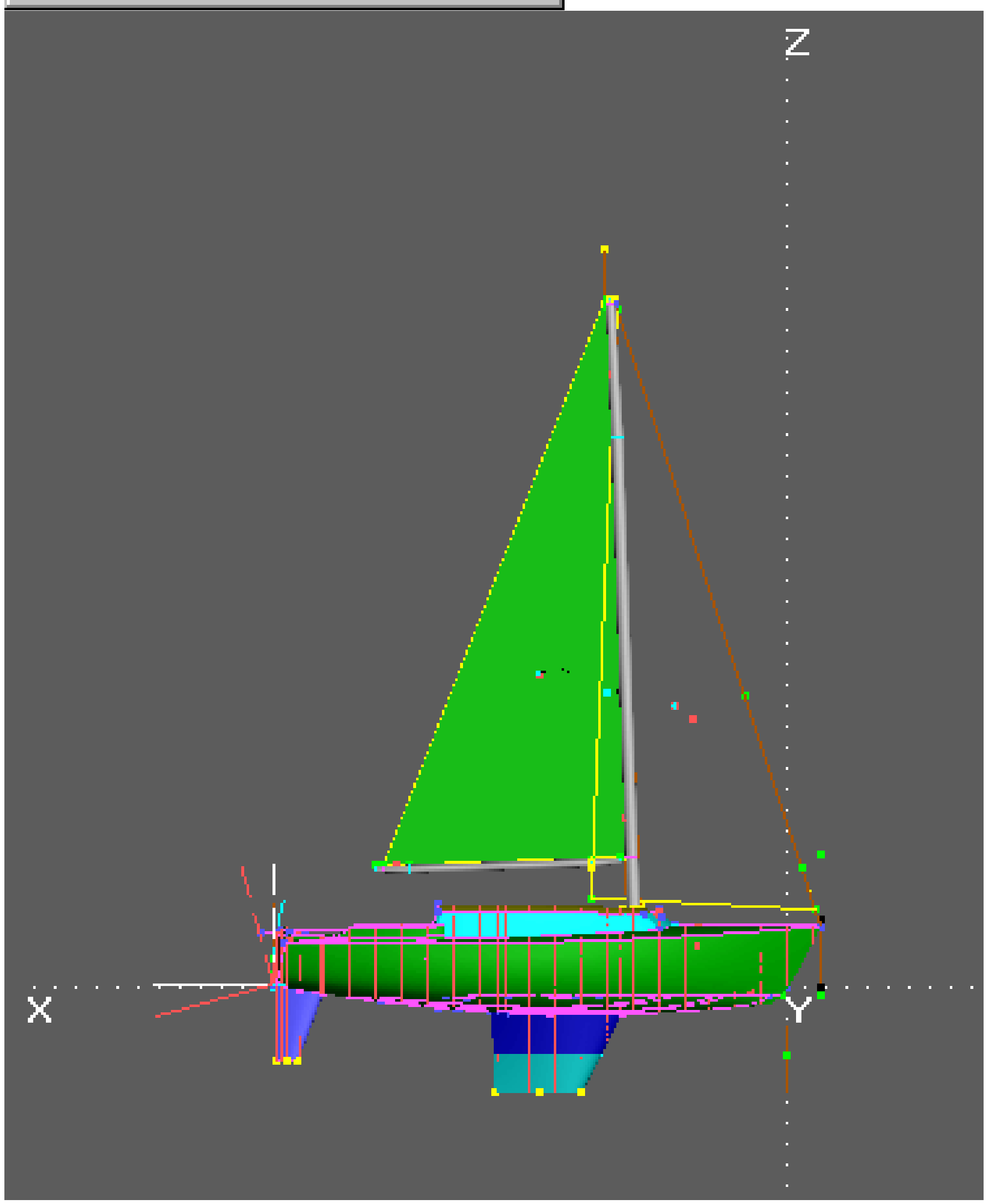
This menu pick or shortcut key displays a Real Values dialog if the following item(s) are in the selection set:

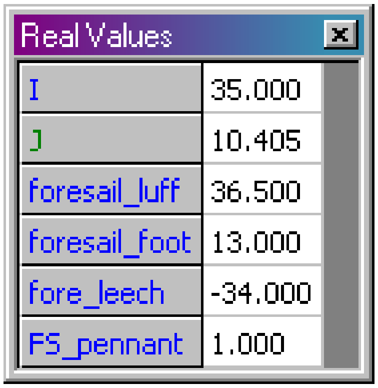
One or more variables
One or more formulas
One or more Entity Lists containing only variables or formulas
Any combination of the above.

In the following images we see a manager displaying the properties of an Entity List named 'sail_dimensions'. This list contains only Variables and / or Formulas. On the right is a display of those reals in the Real Values Dialog. If the name of the real is clicked in the dialog, that real replaces any item in the selection set and now can be edited. If for any reason a value of any real should be changed, either by a direct edit or indirectly from another change, the updated value will be reflected in the Real Values Dialog. Multiple Real Values Dialogs may be open at any time.

NEW In the case of a Variable displayed in a Real Values Dialog, direct editing of that value is now available. No longer does the value need to be changed in the Properties Manager, but can be edited directly in the Real Values Dialog.


Properties
Entity List
Name
sail_dimensions
Layer
15 - Rig Construction
Lock
False
Entities
(6)
User data


Z
X
Y



Redo

Edit>Redo or **Ctrl+Y**

[Only available after Undo has been used.]

When possible, reverses previous undo actions (unlimited number).

To redo actions that have been undone, from the most recent and going back sequentially:

Click or **Edit>Redo** or press **Ctrl+Y** successively.

Relax Control Point

Tools>Special>Relax Control Point

[Only available when a Point (not Projected, Mirrored or Offset) is selected which is an interior control point of a C-spline Curve]

Removes the selected control point from its child spline, solves the resulting reduced spline, then puts the control point back onto the curve in a new position.

Relaxation Factor:

When performed as described above, the relaxed point will move a certain distance. The Relaxation Factor allows you to move the point a specified fraction of that distance.

Curve to Relax:

If the point selected is a parent of more than one C-spline Curve, this drop down list allows you to specify which curve you want to modify.

Note: If the relaxed point is a parent of more than one curve, all child curves will be affected by the relax operation, but the selected curve will be the one "relaxed to".

Relaxation Constraint:

Allows you to specify one coordinate to keep fixed when the point is relocated, or if 'Normal to curve' is selected, moves the point normal to the curve.

Save as Default Settings:

Clicking on this box saves the Relaxation Factor and Relaxation Constraint as default settings. When Relax Control Point is next used, the dialog opens with the default settings. These defaults hold across sessions.

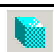
To relax a control point:

1 Click on the point you want to relax, and choose **Tools>Special>Relax Control Point**.

2 Set the Relaxation Factor and Relaxation constraint and choose the Curve to Relax (if more than one is possible).

3 Click OK

Render

 **View>Display>Rendering**

Presents a Rendering Options dialog, then displays a rendered, shaded, lighted view of the model in which surfaces appear as solids with hidden surfaces removed. This view always includes symmetry images. Initially, the rendered view defaults to a perspective viewpoint; but you can change back and forth between perspective and orthographic using the View menu options, toolbar buttons, or keyboard controls (<p>).

The initial rendered view takes its viewpoint from that of the active wireframe, or if invoked from a non-wireframe window, it takes the view of the most recently opened wireframe view. Specifically:

if the wireframe is a perspective view, the initial rendered viewpoint is exactly the same as the wireframe viewpoint

if the wireframe is an orthographic view, the initial rendered image uses the Lat, Lon of the wireframe, but creates a perspective image and does a Zoom All on it

Rendering options

Colors. Wireframe colors or monochrome (radio buttons); if monochrome, specify the color (by number). The program default is wireframe colors.

Lighting directions. Set the latitude and longitude for the pair of rendering light sources. The program default is Latitude = 35, Longitude = 135.

Background color. Specify red, green, blue components with values between 0 and 155 — 0, 0, 0 is black; 255, 255, 255 is white.

Show contours, snakes, curves, and mesh lines. Check boxes toggle display of these lines on / off; the program default is off.

Tansparency. Select from fully opaque to…..invisible or anything in between.

<Make Default>. To customize the default settings for the render options, set up the dialog as you want it, then choose <Make Default>.

To abort displaying a rendered view, just cancel the window (you can do this while the program is drawing).

Rendered view lighting

In the rendered display, the brightness and hue of any region of a surface depends on a combination of:

the surface's intrinsic color

the color of light sources

the surface's orientation with respect to light sources.

SurfaceWorks uses two light sources, both colored white, and of equal brightness. They are infinitely far away, so the light from each source consists of parallel rays. They are always located opposite each other in the global coordinate system, so they shine towards one another from opposite sides of the model. The light direction is fixed with respect to the model; it does not change when you move around the model with the cursor keys. You can control the light source directions by setting the latitude and longitude of one light source in the Render Options dialog. (The second light source will remain opposite.) The initial default direction is latitude 35, longitude 135.

With this lighting system, portions of a surface that are nearly perpendicular to the light direction are bright, and portions which are nearly tangent to the light direction are dark. For most models consisting of curved surfaces, this shading provides a nice level of contrast which makes the surfaces appear attractively three-dimensional. However, a result of the lighting system is that a flat surface is displayed with uniform coloration.

The lighting system also contains some "specular reflectivity", which produces white highlights in areas where the surface is suitably oriented to reflect light in a mirror-like fashion. Occasionally you can get in a position where specular reflection off a flat or nearly flat area dominates the image. This can always be corrected by changing the lighting direction. Producing the most attractive rendered views often requires two stages of setup: (1) moving the camera to the best position with respect to the model, considering framing, proportion and visibility of various model features; then (2) adjusting the light source direction to produce the nicest shading and highlights.

More realistic effects

In a rendered view of SurfaceWorks type, no shadows are cast, and no reflections of one surface in another occur. These more realistic effects require ray tracing techniques, which are significantly more complex and laborious than shaded, lighted views. Ray tracing is not performed in SurfaceWorks, but is available in many add-on graphics packages such as 3D Studio (Autodesk, Inc.) and AccuRender (Robert McNeel Associates).

Reset All Filters

Select>Reset All Filters or **Ctrl+F**

Resets all **class filters** to **on**. **Current Layer Filter** to **off**. **Visible/Hidden/Both** to **Both**.

To reset all class filters:

Choose **Select>Reset All Filters** or **Ctrl+F**.

Rotate

See "Transform>"Rotate" on page 209.

See "Rotate View" on page 209.

Rotate View

View>Modify>Rotate or **Arrow keys**

Allows you to use the mouse to rotate your view of the model. When you choose this option, it changes the cursor into Rotate mode, until you change to another cursor mode (e.g. Pan, Zoom to Area, Select).

To rotate the view of the model:

1. Click , or choose **View>Modify>Rotate**, or **right-click in graphics whitespace** and choose **Rotate** from the shortcut menu.
2. Drag the mouse until you have the view you want.
3. Drag again, as desired (as many times as you want).

To get out of Rotate mode (i.e. to stop twirling around) you can do any of the following:

- Click  or choose **View>Modify>Rotate** again.
- Press the **Esc** (Escape) key.
- **Right-click in graphics whitespace** and choose another cursor mode.
- Choose **another cursor mode** (e.g. Pan, Zoom In/Out, or Select mode) via the toolbar buttons or the View menu.

Save

File>Save or **Ctrl+S**

[Not available in SolidWorks Integration Mode]

Saves your SurfaceWorks modeling as an .ms2 file.

Choose **File>Save**.

Save As

File>Save As

[Not available in SolidWorks Integration mode.]

Saves your SurfaceWorks modeling as an .ms2 file with a new name.

Choose **File>Save As.** Type in new name.

Select All

Select>All or **Ctrl+A**

Clears the Selection Set, then puts into the Selection Set all the entities in the model that can get through the active filters (if any).

To select all entities in the model:

Click  or choose **Select>All** or press **Ctrl+A**.

Select Both

Select>Both

Allows both visible and hidden entities to be selected. **Select>Visible-Hidden-Both** are a set of radio buttons — when you choose one of them (turn it on), you turn the other two off. The choice preceded by the dot is the one that is on.

This is reset to Both each time a model is opened.

Note: Hidden entities cannot be selected in graphics displays, even if you have the Both (or Hidden) filter turned on — you'll need to use the Surface Manager, the Available Entities pane, or **Select>All** instead.

Select By Name

Select>By Name

By selecting this menu item or clicking the toolbar button  a dialog will be accessed for the purpose of selecting an entity or entities by name. You can enter a single name, or multiple entity names separated by spaces. Correct spelling and capitalization are required.

Through the use of wildcards, large groups of entities can be selected at one time.

'*' stands for any set of characters

'?' stands for any single character

For example, 'frame*' would select all the entities in the model which have 'frame' as the first 5 characters of their name. 'pt2?' will select all entities whose names start with 'pt2' and have exactly one additional character.

Select Children

Select>Children

[Only available when at least one entity is selected.]

Adds child entities to the Selection Set. You have the options to add only first generation children or all generations of children. Children are selected even if they are hidden or on turned off layers.

To add the children of one or more entities to the Selection Set via the menus:

1 **Select** the entity or entities whose children you want in the Selection Set.

2 Choose **Select>Children**.

3 Choose either **First Generation** or **All Generations**.

To add first generation children to the Selection Set via the toolbar:

1 **Select** the entity or entities whose children you want in the Selection Set.

2 Click .

First Generation. “First generation” children are the “direct” children of the selected entity(s). These are the children you would see listed on the Surface Manager’s Children tab if you were to expand any of the selected entities. This menu choice adds just the first generation children to the Selection Set.

All Generations. “All generations” includes the first generation children, plus their children, plus their children... This menu choice adds all generations of children to the Selection Set.

Select Expand Entity List

Select>Expand Entity List

This menu item offers two choices: First Generation, or All Generations. The definition of the First Generation option is: Scan the selection set, and if any entity is found to be an Entity List, replace it with the entities in the list. The All Generations option just repeats this procedure until there are no more entity lists in the selection.

Example application: In a complex model, you may have many surfaces that will eventually have to be exported in an IGES file. You may have to repeat this export several times as the design develops. You can maintain a list of the surfaces that need to be exported in an Entity List, named say 'IGES_surfs'. Then when the time comes to export an IGES file version, select 'IGES_surfs', Select>Expand>First Generation, and proceed with File>Export3D>IGES.

Alternatively, you could list your export surfaces in multiple Entity Lists such as 'IGES_hull', IGES_keel', 'IGES_deck', and put these lists into an Entity List named 'IGES_surfs'. Then to select all these surfaces for export, select 'IGES_surfs', and Select> Expand>All Generations.

Select Hidden

Select>Hidden

Constrains selection to hidden entities only. **Select>Visible-Hidden-Both** are a set of radio buttons — when you choose one of them (turn it on), you turn the other two off. The choice preceded by the dot is the one that is on.

This is reset to Both each time a model is opened.

Note: Hidden entities cannot be selected in graphics displays, even if you have the Hidden (or Both) filter turned on — you'll need to use the Surface Manager, the Available Entities pane, or **Select>All** instead.

Select Menu

See "Select Mode" on page 212.
All See "Select All" on page 210.
"Select Parents" on page 212
"Select Children" on page 210
Parents See "To add first generation parents to the Selection Set via the toolbar:" on page 213.
Children See "To add first generation children to the Selection Set via the toolbar:" on page 211.
"Select By Name" on page 210.
"Clear Selection Set" on page 139.
"Invert Selection Set" on page 188
"Select Expand Entity List" on page 211
See "Class Filters" on page 137.
See "Current Layer Filter" on page 155.
See "Reset All Filters" on page 208
Visible See "Select Visible" on page 213.
Hidden See "Select Hidden" on page 211.
Both See "Select Both" on page 210.

Select Mode

Select>Select Mode or **Esc**

[Only available in graphics and Curvature Profile views.]

Changes the cursor and mouse to Select mode from any of the other cursor modes, such as Rotate, Pan, or Zoom to Area.

To turn Select mode on:

Click , or choose **Select>Select Mode,** or press **Esc**.

Select Parents

Select>Parents

[Only available when at least one entity is selected.]

Adds parent entities to the Selection Set. You have the options to add only first generation parents or all generations of parents. Parents are selected even if they are hidden or on turned off layers.

To add the parents of one or more entities to the Selection Set via the menus:

1. **Select** the entity or entities whose parents you want to add to the Selection Set.

2. Choose **Select>Parents**.

3 Choose either **First Generation** or **All Generations**.

To add first generation parents to the Selection Set via the toolbar:

1 **Select** the entity or entities whose parents you want to add to the Selection Set.

2 Click .

First Generation — “First generation” parents are the “direct” parents of the selected entity(s). These are the parents you would see listed if you were to open the Insert or Edit dialogs for the entities, or if on the Surface Manager’s Parents tab you were to expand any of the selected entities. This menu choice adds just the first generation parents to the Selection Set.

All Generations — “All generations” includes the first generation parents, plus their parents, plus their parents... This menu choice adds all generations of parents to the Selection Set.

Select Visible

Select>Visible

Constrains selection to visible entities only. **Select>Visible-Hidden-Both** are a set of radio buttons — when you choose one of them (turn it on), you turn the other two off. The choice preceded by the dot is the one that is on.

This is reset to Both each time a model is opened.

Selection

See “Clear Selection Set” on page 139.

See “Invert Selection Set” on page 188.

See “Zoom to Selection” on page 237.

Session Options

Tools>Options, Session

Options in this tab can be set each time you open a model, but return to the defaults when you close the model.

To change session options:

1 Choose **Tools>Options**.

2 Click the **Session** tab.

3 Make any desired changes.

4 Click **OK**.

Accurate Mode

The Accurate option was originally included for calculating output that would be used for actual construction, e.g. full-scale patterns or N / C (numerical-control) cutting. In practice, we have found that we rarely use Accurate mode because construction-accurate output is readily available simply by increasing the subdivision of surfaces using the Divisions multiplier in Tools>Options> Performance.

When exporting a file, you are asked if you want to do so in Accurate mode and are given the option of having the model stay in Accurate mode while it remains open. This is the equivalent of setting this check box.

What does Accurate mean? SurfaceWorks functions at two levels of accuracy. In normal operation, many calculations are performed quickly by linear interpolation in lookup tables. In Accurate operation, all the direct calculations implied in the logical model are actually performed and the results should be accurate to the full precision displayed (.001 unit), but because of this, Accurate mode is SLOW! In normal mode, by increasing divisions, you can approach Accurate mode accuracy, and in some cases actually attain it. In fact, though, you don't need to get there — there is enough inaccuracy in the physical construction process that Accurate mode accuracy is rarely (if ever) achieved outside of SurfaceWorks.

Set View

View>Modify>Set View

Gives you explicit control of the camera location and viewing direction. This is to allow for visibility problems, e.g. what will you be able to see looking forward from the steering position? or to get the camera back to a specific position for reproducing a rendered view. When you use View>Set View, you effectively set a whole new orbit sphere for the camera.

In the Set View dialog box, you specify:

Camera. Set the (X, Y, Z) position of the camera in the global coordinate system.

Lat. Set the latitude, in degrees, of the viewing direction.

Lon. Set the longitude, in degrees, of the viewing direction.

Radius. Set the distance from the camera to the center of the camera's orbit sphere.

Lens mm. Set the effective focal length of the camera lens. 42 mm is the default lens when a view is opened. A shorter focal length is like a closeup (30-35 mm) or fisheye (<30 mm) lens on a 35 mm camera, producing progressively stronger perspective. A longer lens is like a telephoto, which enlarges the image but reduces perspective. (Orthographic views correspond to the limit of infinite focal length.) <Home> restores the lens to the default 42 mm.

Tilt. Set the tilt of the camera sideways while keeping it aimed at the same look point. This action is like tilting (not twisting) your head from side to side — you see things from an angled perspective (relative to the vertical). Specify tilt in degrees:

- positive tilt rotates the camera counterclockwise (the image rotates clockwise)
- negative tilt rotates the camera clockwise (the image rotates counterclockwise)

The default values offered in the Set View dialog are those of the current viewing status.

If you specify a new camera location (X, Y, and / or Z), you shift both the camera and its orbit sphere, and you therefore also define a new point at which the camera is aimed and around which the camera will rotate. If you pick a small radius (say 0.1 ft), the action of the right and left arrow keys is like standing with your eye at the camera point and turning your head right and left.

Additional Viewing Options

While working in wireframe view there is sometimes a need to set the view parameters so the viewing direction is either normal (perpendicular) to, or parallel to, some geometric element. For example, suppose you have constructed a window on a flat cabin surface that is not parallel to any of the coordinate planes. In order to extract a CAD drawing of the window outline for manufacturing, you want to set a viewing direction normal to the plane of the cabin side surface.

View>Modify>Set View>Normal To allows setting the view "normal to" a variety of selected entities:

- A Plane
- Any three points ("normal to" means normal to the plane of the three points)
- A Frame (normal to any of the 3 coordinate planes of a Frame)
- A Magnet or Ring ("normal to" means normal to the host surface, at the location of the magnet or ring)
- A Curve (with the exception of a straight line) or Snake that is at least approximately planar ("normal to" means normal to the plane of the curve or snake)
- A Surface that is at least approximately planar ("normal to" means normal to a plane that is least-squares fit to the tabulation of the surface.)

View>Modify>Set View>Along allows setting the view "along" a variety of selected entities:

- A Line
- Any two points ("along" means along the line connecting them)
- A Plane (along some direction parallel to the plane)
- Bead or Ring ("along" means along the tangent to the supporting curve or snake, at the location of the bead or ring)
- A Frame (along any of its 3 axes)

Shaded

View>Display>Shaded

[Only available in graphics and Curvature Profile views.]

This is a toggle with Wireframe (the model is always either in Shaded or Wireframe view). Displays a rendered, shaded, lighted image of the model in which surfaces appear as semi-transparent solids (unless transparency is zero). The default transparency is 50%. When a shaded image is displayed in the active window, the Shaded option on the **View>Display** menu is preceded by a checkmark.

Models open in the view they were last saved in.

To display a shaded image:

Click  or choose **View>Display>Shaded**.

To open additional windows for the model:

Choose **Window>New Model Window**.

Multiple windows (displaying Shaded and / or Wireframe images) with different viewing angles can be handy for visually checking the impact of an edit from several directions at once.

To return to a particular graphic window, do either of the following:

- **Click in it**.
- **Select** the window from the open windows list on the **Window** menu.

The Shaded Display Image

In a shaded display, the brightness and hue of any region of a surface depends on a combination of:

- the surface’s intrinsic color
- the color of light sources
- the surface’s orientation with respect to light sources
- the shaded display transparency setting

MultiSurfSurfaceWorks uses two light sources, both colored white, and of equal brightness. They are always located opposite each other in the global coordinate system, so they shine towards one another from opposite sides of the model. The light direction is fixed with respect to the model; it does not change when you rotate, pan, or zoom.

With this lighting system, portions of a surface nearly perpendicular to the light direction are bright, and portions nearly tangent to the light direction are dark. For most models consisting of curved surfaces, this shading provides a nice level of contrast which makes the surfaces appear attractively three-dimensional. However, a result of the lighting system is that a flat surface is displayed with uniform coloration.

The lighting system also contains some “specular reflectivity,” which produces white highlights in areas where the surface is suitably oriented to reflect light in a mirror-like fashion.

Note that in a shaded display of this type, no shadows are cast, and no reflections of one surface in another occur. These more realistic effects require ray tracing techniques, significantly more complex and laborious than shaded, lighted displays. Ray tracing is not performed in SurfaceWorks, but is available in many add-in graphics packages such as PhotoWorks (SolidWorks add-in by LightWork Design Ltd.), 3D Studio (Autodesk, Inc.) and AccuRender (Robert McNeel Associates).

Show

Show-Hide>

 **Show Selection Set** or **Ctrl+W**  **Show Parents**  **Show All**

 **Show Points**  **Show Children**

Here are some quick show options that operate on specific groups of entities. **All** and **Points** ignore the contents of the Selection Set.

Selection Set — makes the entire Selection Set visible.

Points — makes all points in the model visible.

Parents — makes the first-generation parents of the selected entity(s) visible.

Children — makes the first-generation children of the selected entity(s) visible.

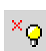
All — makes all entities that are on turned on layers visible.

To show the Selection Set:

1 **Select** the entities you want to show.

2 Click  or choose **Show-Hide>Show Selection Set** or press **Ctrl+W**.

To show all Point entities:

Click  or **Show-Hide>Show Points.**

To show Parents or Children:

1 **Select** the entity or entities whose parents or children you want to show.

2 Click  or **Show-Hide>Show Parents**; or click  **Show-Hide>Show Children.**

To show All:

Simply click  or **Show-Hide>Show All**.

Note: These group show actions, especially Parents and Children, can be very powerful tools to speed editing.

Show-Hide Menu

See "Show" on page 217.
See "Hide" on page 178.

SLDCRV

See “ ”Import .SLDCRV” on page 184

Spline

See “ ”Quick Spline Mode” on page 203

Status Bar

View>Status Bar

Toggles display of the Status Bar on and off. For information about contents of the Status Bar, see “SurfaceWorks Fundamentals - The Status Bar.”

To display the Status Bar:

Choose **View>Status Bar** to toggle the Status Bar on (checked) and off (unchecked).

Surface

See “Flatten” on page 175.

See “Surface Curvature” on page 218.

See “Surface Curvature Options” on page 220.

Surface Curvature

 **View>Display>Surface Curvature**

[Only available when only a single surface entity is selected.]

Displays a color-coded map of normal, Gaussian, or mean surface curvature for the selected surface. The surface curvature display opens in a separate window. You can have multiple surface curvature windows open at the same time.

To display surface curvature:

1. Select a **surface** entity.
2. Click  or choose **View>Display>Surface Curvature**.
3. In the options dialog (**Tools>Options>Surface Curvature tab**), choose **Normal**, **Gaussian**, or **Mean**.
4. If you chose **Normal**, specify the **Angle from u-direction**.
5. Click **OK**.

To open additional surface curvature windows:

1. Make a graphics window active.
2. **Select** a surface (if not already selected).

3 Click  or **View>Display>Surface Curvature**.

To change the color range of an open surface curvature display:

1 From an open Surface Curvature window, choose **Tools>Options** or **right click** and **choose View Options**.

2 The Options open to the **Surface Curvature** tab (the Color Range options are now available).

3 Change the **Color Range** values.

4 Click **OK**.

Curvature Type

Specify the type of curvature display you want to see: normal, Gaussian, or mean (radio buttons). “Angle from u-direction” is required only for Normal curvature. Specify the angle, in degrees, from -90 to 90. The program default is 0 degrees.

For definition and discussion of normal, Gaussian, and mean surface curvature, see “Editing Models - Fairing Surfaces”.

Reading the Surface Curvature Display

In the surface curvature display, curvature of the surface is displayed as a shaded rainbow, each color representing a range of curvature values, from blue (lowest value) through yellow to red (highest value). A column of color-coded curvature values is displayed down the left edge of the screen as a key to the color coding. If white lines appear on the surface that look like contour lines, they show the areas of zero curvature. In other words, they mark the transition between negative and positive curvature.

Curvature intensity is the absolute value of surface curvature. For example, if surface curvature ranges:

- from -3 to +1, the most intense curvature is negative curvature of -3
- from -1 to +3, the most intense curvature is positive curvature of +3
- from -3 to +3, there are two areas of most intense curvature, one negative and one positive, at -3 and +3 respectively

The surface’s positive normal direction is indicated by a white line originating near the center u = .5, v = .5 of the surface (if you can’t see it, change the viewpoint until it appears).

Sometimes a surface can have a small area of intense negative or positive curvature, such that color variation in the default display is confined to that small region, but the majority of the surface appears a fairly uniform color. This can obscure subtle but important variations of curvature in the rest of the surface.

Color Range

The first time you choose Surface Curvature for a surface, the Color Range fields for High and Low are grayed. SurfaceWorks produces the first display showing the entire range of curvature for the surface, with areas of highest curvature in red and areas of lowest curvature in blue. Curvature values in-between are shown in rainbow color gradients progressing from blue (lowest curvature) through yellow to red (highest curvature). For details on changing the values for High and Low in this dialog, see "To change the color range of an open surface curvature display:" on page 219 and "Color Range" on page 220.

Tips for Narrowing the Color Range

To focus in on the area(s) of less intense curvature and better see the curvature variations there, use the **Tools>Options>Surface Curvature>Color Range** settings to narrow the range of curvature to be displayed. Areas having curvature greater than the High value you enter will appear red, and areas having curvature less than the Low value you enter will be blue. The graduation in color from red through yellow to blue now depicts only the variations in curvature between the Low and High values you entered. The two pictures together (default and narrowed curvature range) can give you a much better picture of curvature variation over the entire surface if most of it appears one color in the default display.

For a surface curvature display that comes out mostly bright blue (low curvature) with the default Color Range, we would want to narrow in on that curvature range — so we would narrow the curvature range by making the High value in the dialog smaller, closer to the blue value. For a surface curvature display that comes out mostly yellow in the default display, we would want to make the High value in the dialog smaller (closer to the current value for yellow) and the Low value higher (again, closer to the current yellow value). Generally, you can pick new values to try right off the color code scale at the left of the display.

Coordination of Divisions

If your curvature display looks ripply in a predictable sort of way, it's a sign that you need to either increase the number of divisions on your surface or coordinate divisions between the surface and its parents, in either case enabling more accurate curvature calculations.

It is very helpful to coordinate the divisions of surfaces with the divisions of their parent curves. For example, for a lofted surface, use the same number of divisions on each of the master curves, and also make sure that u-divisions x u-subdivisions is the same as, or a multiple of, the master curve t-divisions x t-subdivisions (e.g.: u = 8x4 and t = 8x4 — surface and curve divisions are the same; or u = 8x4 and t = 4x1 — surface divisions are a multiple of curve divisions). Without this kind of coordination, surface curvature distributions may appear falsely erratic, with complex color variations, because of "aliasing". (Coordination of divisions also produces a more accurate model, in general.)

Surface Curvature Options

Tools>Options>Surface Curvature

[Only available when a surface curvature window is active. This tab also automatically is presented when you select a surface and choose View>Display>Surface Curvature.]

To change surface curvature options:

1 Make the Surface Curvature window active

2 Choose **Tools>Options** or **right click** and **choose View Options**.

3 The Options open to the **Surface Curvature** tab.

4 Make any desired changes.

5 Click **OK**.

Curvature Type

Specify the type of curvature display you want to see: normal, Gaussian, or mean (radio buttons). “Angle from u-direction” is required only for Normal curvature. Specify the angle, in degrees, from -90 to 90. The program default is 90 degrees.

For definition and discussion of normal, Gaussian, and mean surface curvature, see “Editing Models - Fairing Surfaces”.

Color Range

The first time you choose Surface Curvature for a surface, the Color Range fields for High and Low are grayed. SurfaceWorks produces the first display showing the entire range of curvature for the surface, with areas of highest curvature in red and areas of lowest curvature in blue. Curvature values in-between are shown in rainbow color gradients progressing from blue (lowest curvature) through yellow to red (highest curvature). For details on changing the values for High and Low in this dialog, see “To change the color range of an open surface curvature display:” on page 219 and ”Color Range” on page 220.

Symmetry Images

View>Symmetry Images or **F5**

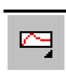
[Only available when a model has Mirror and / or Rotational Symmetry as specified in the Tools>Options>Model dialog.]

Toggles on / off the display of the model's symmetry images. Resets to off each time the model is opened.

To display the model's symmetry images:

Choose **View>Symmetry Images** or press **F5**.

Tangency Profile

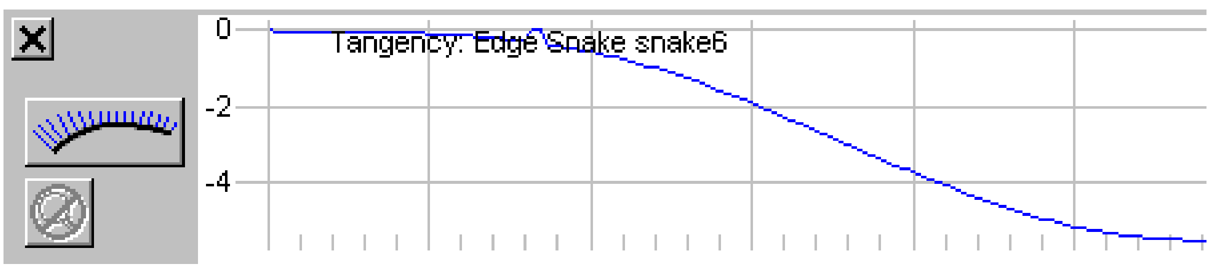


View>Display>Profiles>Tangency

Design problems often arise where it is desired to make one surface approximately or exactly tangent to another surface. This can occur with two surfaces joining edge-to-edge, or where one surface edge is constructed as a snake on another surface. It is generally difficult to tell in either wireframe or render view how accurately this tangency has been achieved.

One way that has been available to make a quantitative measure of tangency is the following construction. Suppose the edge of surface 'B' is intended to be tangent to surface 'A' along some snake 'N' that lies in surface A. Make an Ring 'r1' somewhere on 'N', and an Offset Point 'opA' from 'r1', with an offset.of 1 unit. Make a Projected Magnet 'pm' from 'r1' projected onto 'B', normal to 'B'. (The distance of projection will be essentially zero). Make a second Offset Point 'opB', 1 unit from 'pm'. Then you can use Tools / Measure / Angle with 'opA', 'r1' and 'opB' selected, to measure the difference in direction of the normal vectors to the two surfaces. You can move 'r1' to different places along 'N', and repeate the Angle command, to see how the angle varies.

View / Display / Profiles / Tangency essentially automates this construction at many points along the junction, and displays the result as a graph, similar to Curvature Profile. Tangency Profile is enabled on the menu when one snake and one surface are selected. If you don't already have a snake along the junction, an Edge Snake is the obvious choice.



In this example, the tangency is being measured along an Edge Snake 'n0'. The angle is accurately zero at both ends of 'n0'. Along the length of the snake, the angle varies from slight positive values (less than 0.1 degree) near t = 0.10, to approximately -1.4 degrees near t = 0.50.

Tile Horizontal

Window>Tile Horizontal

Sizes and arranges open SurfaceWorks windows into non-overlapping panes by fitting them to the SurfaceWorks workspace horizontally (one above the next) — with two or three windows open, the panes stretch the full width of the program window; with four or more windows open, the panes are half the width of the program window. The active window is at the top (or top left).

Tile Vertical

Window>Tile Vertical

Sizes and arranges open SurfaceWorks windows into non-overlapping panes based on fitting the SurfaceWorks workspace vertically — with two or three windows open, the panes stretch the full height of the program window; with four or more windows open, the panes are half the height of the program window. The active window is at the left (or top left).

Toolbars

View>Toolbars

Lets you display or hide the SurfaceWorks toolbars, in any combination. For details about each of the toolbars, see "SurfaceWorks Fundamentals - Toolbars and Toolbar Buttons".

Toolbars displayed and their location are application settings.

To display or hide toolbars:

1. Choose **View>Toolbars**.
2. **Check** the toolbars you want on; **uncheck** the ones you want off.
3. Change the Tooltips setting, if desired.
4. Click **OK**.

Show Tooltips. Tooltips are floating fields that appear when the cursor is held over a toolbar icon, to explain its use. To turn Tooltips on, click the Tooltips checkbox. To turn them off, uncheck the box. The initial program default is Tooltips on.

Tools Menu

Measure See "Angle" on page 134.
See "Clearance" on page 138.
See "Distance" on page 159.
See "View Distance Reports" on page 229.
See "Layers" on page 188
See "Mass Properties" on page 190 (Available in SurfaceWorks standalone, and in SolidWorks Integrated mode with MultiSurfSurfaceWorks Marine).
See "Weight Schedule" on page 231 (Available in MultiSurfSurfaceWorks standalone, and in SolidWorks Integrated mode with MultiSurfSurfaceWorks Marine).
See "Hydrostatics" in "Marine" chapter
Special See "Relax Control Point" on page 206.
"B-spline Curve Fit" on page 136
Options See "Tools Options" on page 224.
Command Window See "Commands" chapter.
Entity Transfer (SolidWorks Integration only) See "Using MultiSurfSurfaceWorks with SolidWorks - Reference Guide to MultiSurfSurfaceWorks for SolidWorks Functions in MultiSurfSurfaceWorks - Entity Transfer"
Flatten See "Flattener" chapter.
Hydro See "Hydro" chapter
Macros See "Macros Dialog" in the "Macros" chapter

Tools Options

Tools>Options

Presents program options in a tabbed dialog. The following tabs are provided under Tools>Options. (Surface Curvature is only available when a Surface Curvature window is active; this tab also automatically is displayed when you select a surface and choose View>Display>Surface Curvature):

General:	See "General Options" on page 175.
Display:	See "Display Options" on page 158.
Dragging:	See "Dragging Options" on page 161.
Performance:	See "Performance Options" on page 198.
Entity:	See "Entity Options" on page 163.
Session:	See "Session Options" on page 213.
Model:	See "Model Options" on page 192.
Surface Curvature:	See "Surface Curvature Options" on page 220.

Transform

Edit>Transform

There are three transform options: Scale, Shift, and Rotate.

Locked status and scaling, shifting and rotating

The **Edit>Transform** options to scale, shift, and rotate have the potential to change the attributes of entities, depending on the entities and the transformation selected. When you choose a Transform option and any of the selected entities is locked, you will get a message listing the locked entities, and you will be offered the option to have the program temporarily unlock the entities, perform the specified transformation, then relock all the entities which previously were locked.

Point entities only

Except for Scaling Contours entities, and Scaling and Shifting XYZ Beads and XYZ Rings, Transform only works on Point entities so if you want to transform a curve or surface you must place the curve's or surface's parent points into the Selection Set. One easy way to do this is to select the entity you want to transform and then select its parents to place the parent points into the Selection Set.

All entities that will not explicitly transform are listed in a warning box that comes up when you OK the transformation. This should give you some idea of what entities to check after the transformation.

Always check results

After applying one of these transformations you should always check your results. Some entities do not transform very well, or some will Scale well but not Shift well. If your model looks funny, the problem will most likely have to do with a default entity as a parent. You may have to create some new entities in order make parents that will transform, and take their children with them.

Scale

You can scale either the entire model or the Selection Set (radio buttons). If there is anything in the Selection Set when you choose the scale option, the default will be "Only transformable objects in the selection set".

Scale applies scale factors sx, sy, sz separately to the X, Y and / or Z coordinate directions. The dialog box prompts for entry of the scale factors (defaults = 1). All Points in your file (or Selection Set) will have their coordinates or offsets multiplied by the scaling factor you specify. Also, contours that are parallel to one of the coordinate planes will have 'Distance from Mirror / Surface to first contour' and 'Contour spacing' scaled. Nominally, this produces an affine scaling of your entire model — that is, every point (X, Y, Z), not just every point entity, gets transformed to a new point (sx * X, sy * Y, sz * Z). For many models this will be absolutely true.

Scaling exceptions: The following entities, by their nature, do not scale in a true affine fashion unless the three scale factors are the same (uniform scaling):

Any Point with a non-default Frame parent
Arc, C-spline Curve, Helix
Arc Lofted Surface, C-spline Lofted Surface, Revolution Surface, Offset Surface
Also, any entity created using one of these non-scaling curves will not scale affinely.

In addition, the following do not respond to scale at all:

Entity List, Frame, Plane, Relabel, Wireframe

Here is an example. Suppose 'body' is a Revolution Surface with its axis along the Z-axis and its basis curve in the X,Z-plane. Affine scaling of this surface, say with scale factors 1, 2, 1, should produce a body having 2:1 ellipses for cross sections. However, a Revolution Surface is always going to have circles for cross-sections; in fact, in this example, 'body' would be unaffected by the scaling, because neither its axis nor its basis curve is altered.

Note that while a C-spline Curve or C-spline Lofted Surface does not scale precisely affinely when its parent points are scaled, the results are seldom visibly different from true affine scaling of the curve or surface.

Reversing scaling: You can reverse a scaling by applying reciprocal scaling factors. For example, a scaling of 1, 2, 2.5 would be reversed by a second scaling of 1, 0.5, 0.4; or you could use Edit>Undo.

To scale entities:

1 **Select** the entities you want to scale if you do not want to scale the entire model.

2 Go to **Edit>Transform>Scale**.

3 If you placed entities into the Selection Set the dialog box opens with the radio button set to **Only transformable objects in the selection set** but you can still choose to Scale the entire model by clicking the **All transformable objects in the model** radio button. If there were no entities in the Selection Set then you can only Scale the entire model.

4 **Transform factor for X**, **Y** and **Z** axes; **OK.**

Shift

You can shift either the entire model or the Selection Set (radio buttons). If there is anything in the Selection Set when you choose the shift option, the default will be "Only transformable objects in the selection set".

Shift applies offsets dX, dY, dZ to the X, Y, Z coordinates of all Points in the model (or Selection Set) **that use the defaults for both parents**. All other points shift along with their parents. This translates the complete model (or Selection Set) to a new position, parallel to the old one. A dialog box prompts for the three offsets.

You can reverse a shift by applying negative offsets or by using Edit>Undo.

Warning: If you build geometry using the default origin, '*', as a parent (except for the points mentioned above), the geometry will not shift. If you have this situation and you want to shift, you should insert a point at the origin and edit the entities that use the default origin to use this point instead. (Adopt Children will not work with '*'). It is best to always create a point at the origin if you want to use the origin as a parent.

Contours will not shift unless their Mirror/Surface is a shiftable entity. If you think you will shift your model, build your contours from a 2- or 3-Point Plane.

To shift entities:

1 **Select** the entities you want to shift if you do not want to shift the entire model.

2 Go to **Edit>Transform>Shift**.

3 If you placed entities into the Selection Set the dialog box opens with the radio button set to **Only transformable objects in the selection set** but you can still choose to Shift the entire model by clicking the **All transformable objects in the model** radio button. If there were no entities in the Selection Set then you can only Shift the entire model.

4 **Transform factor for** X, Y and Z axes; **OK.**

Rotate

You can rotate either the entire model or the Selection Set (radio buttons). If there is anything in the Selection Set when you choose the rotate option, the default will be "Only transformable objects in the selection set".

Rotate performs a rotation of the Selection Set (or all) Points **that use both default parents or that use a non-default Point parent** about a specified axis. Points that use a frame as their Frame parent rotate with their respective frames. Any entity that uses a default plane as its parent will not rotate. Mirrored, Projected and Intersection entities by themselves will not rotate. As a part of a larger group of entities, Mirrored entities **may** rotate but Projected and Intersection entities still will not, and must be fixed manually after a rotation is performed. The axis can be either an existing Line entity in the model or one of the three coordinate axes (X, Y, or Z). You pick the axis from a list box; if you will be using a line for the axis, you must select it before you choose **Edit>Transform>Rotate** and make it the last entity in the Selection Set. Then you are prompted for a rotation angle. The angle is calculated using a right-hand rule: if you grasp the axis in your right hand, with your thumb pointing along the line toward the t = 1 end of the line or the positive end of the coordinate axis, your fingers will point in the direction of rotation.

Note: Because SurfaceWorks uses relational geometry, Rotation is a surprisingly complex undertaking. You should only rotate if you have no other choice, and should always check the resulting geometry to make sure it is displaying what you want in the new orientation.

You can undo a rotation by using the same axis and the negative of the original angle or by using Edit>Undo.

When a model uses symmetry, the results of rotation can be surprising. For example, DEMO is a sailboat hull model using Y-symmetry. If you try to display DEMO heeled 20 degrees to starboard by making a -20 degree rotation about the positive X-axis, you will see that the starboard side of the hull does indeed heel 20 degrees to starboard, but the port side (which is still formed by reflection in the Y = 0 plane) heels 20 degrees to port.

To rotate entities:

1. **Select** the entities you want to rotate if you do not want to rotate the entire model. If you want to rotate around a line, the line must be placed last in the Selection Set.

2. Go to **Edit>Transform>Rotate**.

3. f you placed entities into the Selection Set the dialog box opens with the radio button set to **Only transformable objects in the selection set** but you can still choose to Rotate the entire model by clicking the **All transformable objects in the model** radio button. If there were no entities in the Selection Set then you can only Rotate the entire model.

4. **Input rotation angle**; **OK.**

Trim Snake Tool

Tools>Trim Snake or **T**

This is an automatic snake trimming operation. Select 2 snakes which cross. The first snake selected will be the Trim Tool and the second the Snake to Trim. Press <T> on the keyboard and the cursor will change to a Scissors. Hover over the snake to trim and you will see a color change wherever you hover. Click the mouse to Trim the highlighted end of the snake. If you need to trim the other snake, simply control click the other snake, press <T> and you repeat the process.

At this time the basis snake and supporting points for the SubSnake are hidden and kept on the current layer.

Undo

Edit>Undo or **Ctrl+Z**

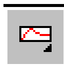
When possible, reverses previous changes you have made (unlimited number).

To undo, from the most recent edit and going back sequentially:

Click  or **Edit>Undo** or press **Ctrl+Z** successively (no limit).

Note: If when creating or editing an entity, you used Apply after specifying one or more properties or parents, Undo will undo in steps, each step delimited by when you clicked Apply. If you made all your entity specifications and then chose OK (or Apply), Undo will undo the entire creation or set of edits at once.

Velocity Profile

 **View>Display>Profile>Velocity**

[Available when one curve or snake is selected.]

The velocity of a curve is the rate of change of distance (arc length) along the curve with respect to change in the t parameter. For example, on an unrelabeled Line or Arc the velocity is constant – equal t-intervals have exactly equal arc lengths. But on a B-Spline Curve the velocity depends on the spacing between control points, being low in regions where two or more control points are close together and relatively high where control points are far apart. On most kinds of curves, velocity is variable with respect to t -- the product of the particular mathematical operations involved in computing the curve.

A discontinuity (jump) in velocity, as can occur especially with PolyCurves and PolySnakes, is generally quite unapparent in a wireframe view, but it's just as much of a challenge for curve fitting as a discontinuity in slope. The chief advantage of a PolyCurve2 (or PolySnake2) is that the velocity is automatically "balanced" at each junction, removing any velocity discontinuities. This makes it much easier to approximate with B-spline (NURBS) math. The PolyCurve2 and PolySnake2 are the default PolyCurves and Polysnakes in SurfaceWorks.

A curve can have zero velocity due to either relabeling or, for some curve types, to the locations of control points. For example, a Relabel that starts with two zero values, e.g. {0, 0, 1} produce zero velocity at the t=0 end of any curve. {0, 0, 1, 1} produces zero velocity at both ends. For another example, a B-Spline Curve that uses the same point twice at either end will have zero velocity at that end. This can happen if you accidentally select the same control point twice, an error which may not be at all apparent in the shape of the curve. Zero velocity on a curve can cause trouble in constructions based on that curve – for example, a Tangent Point won't evaluate if its bead is at a zero velocity point; a Sweep Surface can't have a zero velocity point on its path curve. A curve that has zero velocity at either end may have no fair extension beyond that end.

Turning on tickmarks (visibility option 2) is one way to visualize the velocity distribution of a curve. Velocity is low where the tickmarks are close together, high where they are far apart.

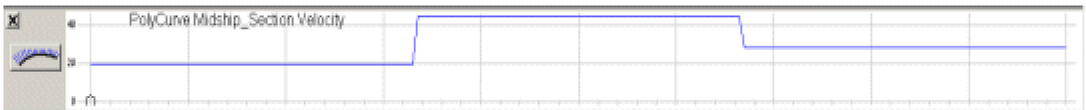
View / Display / Profile / Velocity Profile is a much sharper tool for visualizing velocity. It displays a graph of the velocity distribution, similar in appearance to the Curvature Profile. The differences are:

1 the ordinate plotted is the velocity rather than the curvature

2 the horizontal axis of the graph is just the t parameter. Tickmarks at 0.1 intervals (in the t parameter) assist in the t locations of graph features.

The Porcupine button and command work with Velocity Profile as well as Curvature profile. This provides an alternative presentation of velocity variation, displayed as a set of short lines or “quills” perpendicular to the displayed curve, each with length proportional to the local velocity.

Example: Open sample file POLYCRV1.MS2. Select the PolyCurve ‘midsection’ and activate Tools / Velocity Profile.



The graph indicates that each of the 3 segments between junctions has constant velocity. However, the three velocities are different, so there are discontinuities (shown as steep jumps) at the transition t values of 0.4 and 0.6. Click the Porcupine button; this turns on the Porcupine display, which provides an alternative visual indication of discontinuities in velocity at t = 0.333 and 0.666.

Select the three component curves ‘bottom’, ‘bilge’, and ‘side’ in that order, and create a PolyCurve2. Display its velocity profile; it will be 100% constant.

The applications of Velocity Profile are chiefly diagnostic. Why is this Sweep Surface complaining about the curve I’ve given it for its path? Why does this curve or snake approximate poorly during IGES export? – it looks smooth enough. Why are the longitudinals on my C-Spline Lofted hull compressed in this region? Often the Velocity Profile will provide the answer to these questions.

View

See "Marine - Marine View Orientation".

See “Mechanical View Orientation” on page 191.

Previous See “Previous View” on page 201.

See “View Distance Reports” on page 229.

See "View Menu" on page 230

View Distance Reports

Tools>Measure>View Distance Reports

Toggles on / off the display of all distance reports.

To toggle the display of distance reports:

1 Choose **Tools>Measure>View Distance Reports**. When the menu option is preceded by a check mark, all existing reports will be displayed; when there is no check mark, the display is off. The default when you start SurfaceWorks is on (checked). This is turned on when you enter “distance” mode.

2 To change the display status, just **click the menu item**.

To specify and position reports:

Use **Distance Mode**:  or **Tools>Measure>Distance**.

View Menu

Display

See "Wireframe" on page 235.

See "Shaded" on page 215.

See "Perspective" on page 200.

See "Marine - Ship Lines".

See "Marine - Marine View Orientation".

See "Marine - Offsets".

Profile

See "Clearance Profile" on page 138.

See "Curvature Profile" on page 156.

See "Graph Profile" on page 176

See "Tangency Profile" on page 221.

See "Velocity Profile" on page 228.

See "Surface Curvature" on page 218.

See "Mechanical View Orientation" on page 191.

Modify

See "Previous View" on page 201.

See "Rotate View" on page 209.

See "Pan" on page 197.

See "Zoom In/Out" on page 236.

See "Zoom to Fit" on page 237.

See "Zoom to Selection" on page 237.

See "Zoom to Area" on page 237.

See "Axes" on page 136.

See "Grid" on page 177.

See "Symmetry Images" on page 221

See "Nametags" on page 194.

See "Entity Orientation" on page 164.

See "Toolbars" on page 223.

See "Status Bar" on page 218.

Visible

See "Select Visible" on page 213.

Weight Schedule

Tools>Weight Schedule

[In SurfaceWorks standalone, and in SolidWorks Integrated mode with SurfaceWorks Marine only]

SurfaceWorks entities in the point, curve, surface, snake, and contours classes can have a weight attribute, as follows:

points	weight
curves	weight / unit length
snakes	weight / unit length
contours	weight / unit length
surfaces	weight / unit area

For example, in English units you would specify lb/ft^2 for a surface, lb/ft for a curve, and lb for a point.

SurfaceWorks will then maintain a "weight schedule" or tabulation of weights, centers of gravity, and moments of inertia for all entities that have a nonzero weight attribute, and will compute the total weights, centers, and moments of inertia for the model.

Model-level symmetry, if present, is factored into the results — weight (Wt.) for each entity is the total of its own weight plus the weight(s) of its symmetry image(s). In other words, the entity's weight is doubled for a single mirror symmetry, tripled for a 3-fold rotational symmetry, quadrupled for a 4-fold rotational symmetry or if affected by two mirror symmetries, etc. Therefore, be careful when assigning weights to entities that lie in the reflecting plane of a model-level symmetry — give them a half, a third (or whatever is appropriate) of the total weight.

Tools>Weight Schedule displays the tabulation in a read-only text window, which can be saved as a text file with right click menu>Save As Text from the Weight Schedule window.

Sample output (using DEMO.MS2):

In the DEMO model, only the following entities have weights:

'P11' — 15.70 lb, representing half (the model has Y-symmetry and 'P11' lies in the Y=0 plane) the weight of a stemhead fitting
'MC1', 'MC3' — 2.50 lb/ft, representing structural reinforcement
'hull' — 3.70 lb/ft^2, representing the hull shell
'stations' — 1.20 lb/ft, representing structural ribs

```
Weight Schedule     30-Jul-2001  16:29:37
Model file: DEMO.MS2
Points
  Entity     Object Name    lb       Wt.(lb)     X       Y       Z
  FramePoint  P11           15.70    31.40    0.000   0.000   3.600
Curves
```

```
  Entity     Object Name   lb/ft    Wt.(lb)     X       Y       Z
  BCurve      MC1            2.50    27.92    1.272   0.000   1.133
  BCurve      MC3            2.50    26.32   30.000   0.000   0.851
Contours
  Entity     Object Name   lb/ft    Wt.(lb)     X       Y       Z
  Contours    stations       1.20   165.25   16.236   0.000   0.312
Surfaces
  Entity     Object Name   lb/ft^2  Wt.(lb)     X       Y       Z
  CLoftSurf   hull           3.70  1398.60   15.943   0.000   0.323

              Totals               1649.49   15.645   0.000   0.407
...
```

Weight Schedule Options

Tools>Weight Schedule has several options that allow weight, center of gravity and moment of inertia calculations to be made with subsets of entities.

When no weight schedule is open and Tools>Weight Schedule is selected from the menu, the Weight Schedule Dialog opens.

The radio buttons in the "Entities" group are the principal controls. Each puts the Weight Schedule view into a different mode, which will be maintained until you come back to this dialog and change it. There are five modes, some of which utilize or interact with other controls in the dialog.

"Selected entities" mode

Sometimes it's useful to monitor the weight and c.g. of a single entity, a small group of entities, or a large set of entities that is still a subset of the entire model. An Entity List can be used to store a single list of weight schedule entities that you have selected for this purpose. The Entity List would need to be the only entity in the selection set. The weight schedule report will reflect the contents of the list even though it is the only selected item, i.e. the Entity List is "expanded" (all generations) before use for the weight schedule.

"Current layer" mode

In this mode, the weight schedule is calculated only from entities that are on the current layer (set in the Tools>Layers dialog). If you change the current layer setting while in this mode (for example, from layer 9 to layer 12), the weight schedule will stop showing the entities on the old current layer and start showing the entities on the new current layer.

"Enabled layers" mode

In this mode, the weight schedule is calculated using only entities that are on currently enabled layers (layers that are checked "On" in the Tools>Layers dialog). If you change the choice of which layers are on, the Weight Schedule changes the entities it lists based on your choice.

"Specified layers" mode

In this mode, you can choose any combination of layers, regardless of whether they are on or off. The selection of layers is specified in the edit window labeled "Layers", described below. This selection of layers stays constant when the Weight Schedule is switched to other modes, but has no effect on operation in the other modes.

"Layers" window

The "Layers" control is the key to specifying the set of layers to be used for weight schedule calculation in the "Specified layers" mode. Use single layer numbers or ranges, separated by commas or spaces. Ranges are indicated by the first and last indices, separated by a single hyphen (and no spaces). Duplicates, negative indices, and indices outside the range 0-255 will be ignored.

Examples

3,5 6-12 22

3 5 6-12,22

Both lists indicate layers 3, 5, 6 through 12 and 22.

"Entire model" mode

In this mode, weights are calculated and summed for all entities in the model that have nonzero unit weight, regardless of what layer they are on, or whether their layers are enabled, or are listed in the "Layers" window. ("Entire model" mode is the way Weight Schedule worked in versions of SurfaceWorks before 6.0.)

"Include Moments of Inertia" checkbox

This checkbox makes moments of inertia calculations optional. (While moments of inertia are critical for some forms of analysis -- for example seakeeping, motions in waves -- they are irrelevant for most weight schedule applications.) The default setting for this checkbox is "Off" (unchecked).

OK and Cancel Buttons

"OK" accepts the settings of the dialog, and launches the Weight Schedule view, with the chosen settings. "OK" also causes the program to store the current settings and choices made in the dialog, as the default settings for the next time Weight Schedule is invoked.

"Cancel" bypasses launch of the Weight Schedule view. The dialog simply closes, and any changes in settings that were made in the dialog are discarded.

Automatic updating of the Weight Schedule View

The Weight Schedule is organized as a view, so it updates itself whenever anything in the model changes, even if the Weight Schedule is behind other views. When it updates, it uses the current mode and other specifications that were set in the Weight Schedule Options dialog when the view was last opened.

Empty Weight Schedule

It's entirely possible for the Weight Schedule to open with no entities at all displayed, and zeroes for the weight and c.g. This will happen if no entities are currently selected for the weight schedule to use, that have non-zero Weight attributes. For example:

1 If you are in "Selected entities" mode, and have selected no entities.

2 If you are in "Selected entities" mode, and none of the selected "weight schedule entities" has a Weight attribute.

3 If you are in "Current layer" mode, and there are no entities on the current layer that have Weight attributes assigned.

4 If you are in "Entire model" mode, and there are no entities in the entire model that have Weight attributes assigned.

Weight Distribution File

Whenever a Weight Schedule is generated, a text file named %wt.dist.txt is written in the same folder as the model file. This is the weight distribution file required for longitudinal strength calculations in GHS. (GHS is a marine application).

Printing

Whenever a Weight Schedule is open, the context sensitive menu (right-click) will provide access to a Page Set-up and Print menu choice.

What's Wrong

"What's Wrong" brings up the Errors Manager Dialog listing current model errors (by kind of entity, name, and error code) and error messages to help you fix the problem.

To open the Errors Manager Dialog:

Choose **What's Wrong** from a shortcut menu in any of the following locations:

- In the Surface Manager, Available Entities Manager or Selection Set Manager right-click on an entity marked in error
- Right click in white space anywhere in the graphics window. If there are any entities in error, **What's Wrong** will be available.

Or

- Click on the Errors Manager Dialog button at the bottom of the screen

In the Errors Manager Dialog, you can double-click an entity in the list or select an entitiy and click on the 'Select Entity' button to edit the entity's definition and clear the error. As you fix the errors, the error messages will be cleared from the Errors Manager Dialog.

Don't worry about entities with error 284 "Entity is in error because one or more of its parents is in error." Fixing the entity(s) listed above it that have other errors will get rid of these child-entity errors. If you don't want to see them check the "Hide entities in error due to parent error" checkbox.

Error Details window

The Errors Manager Dialog has a window that reports additional useful diagnostic information about some entity errors. For example, when a Projected Snake which has failed is highlighted in the error view listbox, the Error Details window will display “Projection failed at t = XXX”, where XXX is the t parameter location where the problem occurred. With this clue, you might be able to go to that particular place (for example, put a bead at that t position on the supporting curve) and understand why the projection failed at that location.

As another example, if your Trimmed Surface gets error 317, “Snakes don’t connect up end-to-end to make closed loops”, the Error Details window might display “Snakes 4 and 5 failed to join up; u,v-distance .00241 vs. tolerance .00010.” This reveals the location of the problem, so you can investigate that particular junction and discover why it isn’t sufficiently accurate.

Window

Errors Window, see “What’s Wrong” on page 234.

See “New Model Window” on page 196.

See “Open Windows List” on page 196.

Window Menu

See “New Model Window” on page 196.
See “Cascade” on page 136.
See "Tile Horizontal" on page 222.
See "Tile Vertical" on page 222.
See “Arrange Icons” on page 135.

Wireframe

View>Display>Wireframe

[Not available from a Surface Curvature window.]

Displays the entire visible model as “wires.” It is a transparent image (no hidden lines). When a wireframe image is displayed in the active window, the Wireframe option on the **View>Display** menu is preceded by a checkmark.

Models open in either Shaded or Wireframe view depending on the view as it was last saved.

To display a wireframe image:

Click  or choose **View>Display>Wireframe**.

To open additional work windows for the model (default display is shaded):

1. Choose **Window>New Model Window**.

2. To make the image wireframe, click  or choose **View>Display>Wireframe**.

Multiple windows (displaying shaded and / or wireframe images) with different viewing angles can be handy for visually checking the impact of an edit from several directions at once.

To return to a particular graphic window, do either of the following:

- **Click in it**.
- **Select** the window from the open windows list on the **Window** menu.

Zoom In/Out

View>Modify>Zoom In/Out or <**Pg Up**> and <**Pg Down**> or **center mouse wheel**

Allows you to use the mouse to zoom the view in and out, meaning that as you click and drag the cursor up and down, you appear to zoom in closer to or zoom out farther away from the model. Except when using the center mouse wheel, the cursor changes to while in Zoom In / Out mode.

To use the center mouse wheel, just rotate the wheel and the view will zoom. The cursor shape does not change if you use the wheel.

To zoom the view in and out:

1. Click , or choose **View>Modify>Zoom In/Out**, or **right-click in graphics whitespace** and choose **Zoom In/Out** from the shortcut menu.
2. Press and hold the left mouse button and drag the mouse upward (to zoom in) or downward (to zoom out) until you have the view you want.
3. Drag again, as desired (as many times as you want).

To zoom the view in and out with the center mouse wheel:

1. Rotate the center mouse wheel.

Note: You can also zoom in and out by preset increments using the keyboard: PgUp (“Zoom out”) and PgDown(“Zoom in”).

To get out of Zoom In/Out mode (i.e. to stop zooming) you can do any of the following:

- Click or choose **View>Modify>Zoom In/Out** again.
- Press the **Esc** (Escape) key.
- **Right-click in graphics whitespace** and choose another cursor mode.
- Choose **another cursor mode** (e.g. Rotate, Pan, or Select mode) via the toolbar buttons, the right-click shortcut menu, or the View menu.

Zoom to Area

View>Modify>Zoom to Area or **Shift+center mouse wheel / button**

Zooms to frame the selected rectangular portion of the image. Cursor shape changes to while in this mode.

To zoom in on a particular portion of the model:

1. Click or choose **View>Modify>Zoom to Area**, or **right-click in graphics whitespace** and choose **Zoom to Area** from the shortcut menu to get into Zoom to Area mode, and then hold down the left mouse button at one corner of the area you want to zoom into or hold down the **Shift** key and then **press the center mouse wheel** or **button** at one corner of the area you want to zoom into.
2. **Drag** to the diagonally opposite corner of the area you want to zoom in on.
3. **Release** the mouse button.

To get out of Zoom to Area mode (i.e. to stop dragging zoom boxes with left mouse clicks) you can do any of the following:

- Click or choose **View>Modify>Zoom to Area** again.
- Press the **Esc** (Escape) key.
- Choose **another cursor mode** (e.g. Rotate, Pan, or Select mode) via the toolbar buttons, the right-click shortcut menu, or the View menu.

Zoom to Fit

View>Modify>Zoom to Fit or **F**

Zooms automatically in or out to frame and center the entire model image in the display.

To display the entire model:

Click , or choose **View>Modify>Zoom to Fit**, or **right-click in graphics whitespace** and choose **Zoom to Fit** from the shortcut menu, or press **F**.

Zoom to Selection

View>Modify>Zoom to Selection

Zooms to frame and center the selected entities.

To zoom to the selected entities:

1. **Select the one or more entities** you want to zoom in on.
2. Click , or choose **View>Modify>Zoom to Selection**, or **right-click in graphics whitespace** and choose **Zoom to Selection** from the shortcut menu.

Entity Descriptions

Overview

The following provide descriptions and examples of all the SurfaceWorks entities, in alphabetical order:

Arc

Arc Lofted Surface

...

XYZ Bead

XYZ Ring

Each entity description:

- lists items required to define that particular entity (characteristic properties, such as degree or location, and parent entities)
- describes the roles of those properties and parent entities
- presents one or more examples

Besides the entity-specific properties listed, other properties used to define entities are:

- name, color, visibility, and layer for each entity
- t-divisions and t-subdivisions for curves and snakes
- u- and v-divisions and subdivisions for surfaces

For more details about these data items, see "Understanding SurfaceWorks Entities - Defining Entities".

Some groups of entities share much in common. For instance B-spline Curves, B-spline Snakes, and B-spline Lofted Surface all have a Degree property, and the definition of Degree is the same for all three entities. In such cases, we collect this common information in "Understanding SurfaceWorks Entities - Collective Entity Information".

The Entity Description Examples

The entity description examples include the filename of the sample model file, a figure with some of the key entities labeled, and a brief discussion.

Note: There is information in the examples that is not repeated elsewhere in the documentation.

The example model files are located in the \SurfaceWorks\Examples folder. (If you have SurfaceWorks **for SolidWorks**, you have the option of using the .sldprt example files found in the \Integrated Examples folder). We suggest you load and explore them using the Surface Manager, **View>Nametags**, and **Edit>Definition** to help you identify the entities and relationships in each model. Because many of the examples are taken from real life models, you may have to rotate them in order to see what we are trying to show.

Note: When you close the sample models, do NOT save any changes you may have made during your explorations.

3-point Frame

Parents and characteristic properties

Orientation = Right-handed or Left-handed
Type = 0 to 5
Point1 = Origin of the frame.
Point2 = The primary axis is in the direction from point 1 to point 2
Point3 = The secondary axis is in the plane of the 3 points, with point3 at a positive secondary coordinate.
Display size = is a real parameter with units of length.

Description

Orientation: right-handed frames are conventional and much more commonly used. Occasionally there's a situation where it's more convenient or natural to use a left-handed frame.

Type: 3-point frame types are:

type-0	point1->point2 is x axis, point3 in x, y plane, at positive y
type-1	point1->point2 is x axis, point3 in x, z plane, at positive z
type-2	point1->point2 is y axis, point3 in y, z plane, at positive z
type-3	point1->point2 is y axis, point3 in x, y plane, at positive x
type-4	point1->point2 is z axis, point3 in x, z plane, at positive x
type-5	point1->point2 is z axis, point3 in y, z plane, at positive y

When Display size is positive, it directly controls the lengths of the 3 axes in a wireframe display. If display-size is 0 or negative, the frame is sized as a fixed fraction (1/5) of the model bounding box size.

For more details about specifying Frame entities see Frame Entities in the Understanding SurfaceWorks Entities chapter.

Example

Frame3Pts1.ms2 This model has several Points, and 2 Frames. (Turn on Point Nametags.)

Frame F0 (bright white) is made from Points p1, p2 and p3, in that order. Its origin is at p1. It is type 0, so its primary axis, in the direction from p1 to p2, is x; and its secondary axis, in the plane of the 3 points, is y (with p3 in positive y). Its display size property is 0, so it sizes itself to 20% of the model bounding box. (The model bounding box extents are 2, 2, 0, average 1.333, so the displayed axes of F0 are 0.266 long.)

Frame F1 (yellow) is made from Points p3, p5 and p4, in that order. Its origin is at p3. It is type 4, so its primary axis, in the direction from p3 to p5, is z; and its secondary axis, in the plane of the 3 points, is x (with p4 in positive x). Its display size property is 0.5.

3PointFrame.ms2 An example of an application of a 3 Point Frame. The frame is rotated to facilitate the proper location of a mirror.

See also: RPY Frame

Arc

Parents and characteristic properties

Type = one of 6 types of arc
Point1 = point entity
Point2 = point entity
Point3 = point entity

Description

The Arc is a circular arc passing through Point1. The arc lies in the plane containing the 3 points. If the 3 points lie on a line, the result will be a straight line segment instead of a circular arc. There are 6 types of Arc:

- Start at Point1, through Point2, end at Point3

  the arc passes through all 3 points

- From Point1, with Point2 as center, to Point2-Point3 line

  the arc starts at Point1, uses Point2 for its center, and ends (at the correct radius) on a line from the center through Point3

- Full circle from Point1, Point2 as center, in plane of 3 points

  the arc is a complete circle starting at Point1, using Point2 for its center, and lying in the plane of the 3 points; Point3 is used to set the direction of the circle — the closest point on the circle to Point3 is at a parameter value t that is less than 0.5

- Start at Point1 tangent to Point1-Point2 line, end at Point3

  the arc starts at Point1, uses Point2 for the middle point of its polyline, ends at Point3, and is tangent to its polyline at the t=0 end

- Start at Point1, end at Point3 tangent to Point2-Point3 line

  the arc starts at Point1, uses Point2 for the middle point of its polyline, ends at Point3, and is tangent to its polyline at the t=1 end

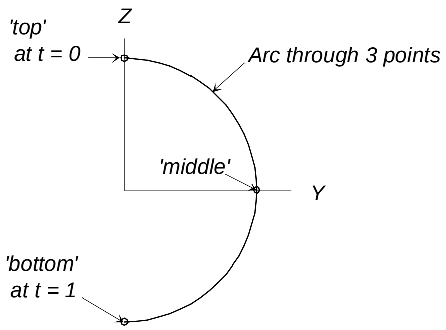
- Semicircle from Point1, oint2 as center, in plane of 3 points

  the arc is a semi-circle starting at Point1, using Point2 as its center, and lying in the plane of the 3 points; Point3 is used to determine which of the two possible semi-circles you get — the one that passes closest to Point3

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is uniformly distributed with respect to arc length or angle.

Example 1: Arc1

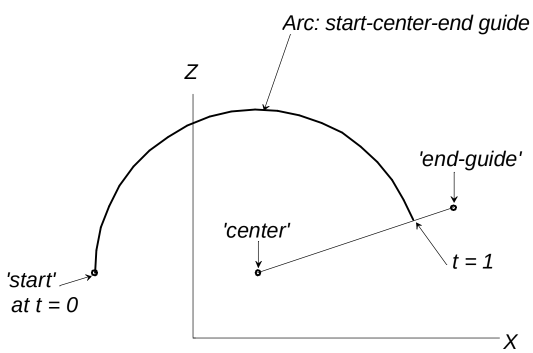
'midsection' is an "Start at Point1, through Point2, end at Point3" of 180 degrees, lying in the X = 0 plane. This type of arc passes through all three of its control points: 'top', 'middle', and 'bottom'.



Arc1

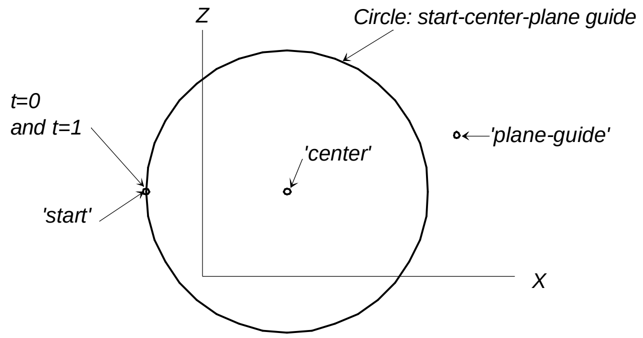
Example 2: Arc2

'arc_to_edge' is an Arc of type "From Point1, with Point2 as center, to Point2-Point3 line" that begins at the point 'start', has its center at the point 'center', and ends at the correct radius on the line from 'center' to 'end_guide'. The arc lies in the plane of the three points.



Arc2

Example 3: Arc3



Arc3

'full_circle' is an Arc of type "Full circle from Point1, Point2 as center, in plane of 3 points." It begins and ends at 'start', uses 'center' for its center, and lies in the plane of the three points.

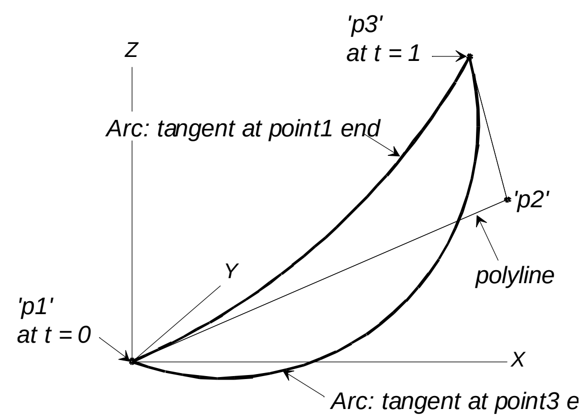
Example 4: Arcs4-5

This model shows the two arc types that are tangent to their polyline at one end. The polyline is the same for both arcs since the two arcs use the same 3 control points, in the same order.

'arc4' (green), "Start at Point1 tangent to Point1-Point2 line, end at Point3," is tangent to the polyline at point1, or the t = 0 end.

'arc5' (cyan), "Start at Point1 tangent to Point1-Point2 line, end at Point3," is tangent to the polyline at point3, or the t = 1 end.

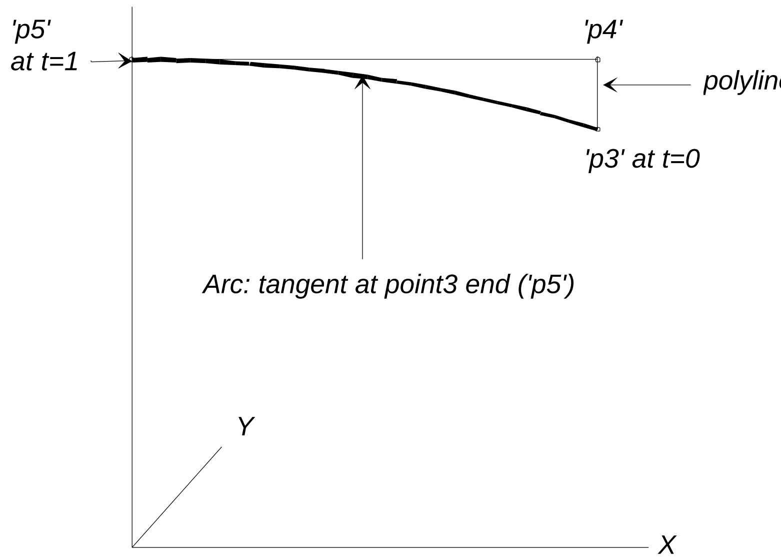
These tangencies are preserved if you move the control points.



Arcs4-5

Example 5: Arc5perp

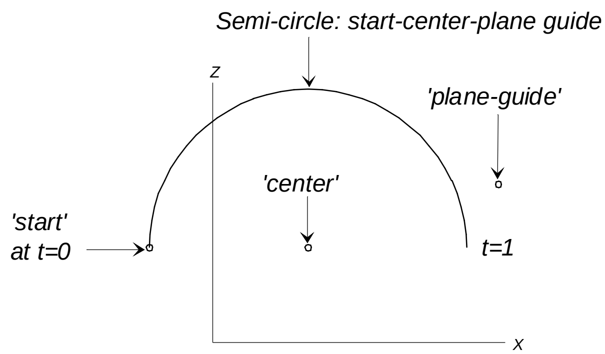
This example shows how to construct an arc that ends perpendicular to a plane: 'p5' (cyan) is a Projected Point, therefore the polyline from 'p4' to 'p5' is perpendicular to the *X = 0 centerplane. The arc 'arch' (blue) is tangent to this polyline at its third control point (t = 1) end.



Arc5perp.

Example 6: Arc6

'semi_circle' is an arc of type "Semicircle from Point1, oint2 as center, in plane of 3 points," made from the same set of points as the "Full circle from Point1, Point2 as center, in plane of 3 points," in Example 3. Notice that of the two semi-circles possible, the one we get is the one that passes closest to 'plane_guide'.



Arc6

See also: Arc Lofted Surface, Helix

Arc-length Bead

Parents and characteristic properties

Bead/curve = parent curve
Arc dist = arc-length distance along curve measured in the units of the model

Description

An Arc-length Bead is a bead positioned along a curve by arc-length distance (measured in the units of the model as opposed to parameter-t distance or X-, Y-, or Z-coordinate). The Arc-length Bead entity is specified by naming the Bead or Curve and the arc-length distance along that curve (or arc-length distance offset from another bead on the same curve). The Arc-length Bead entity can be used wherever a point or bead is called for.

Curve can be any kind of curve (including snakes). When Relative to is checked, the Curve field will gray out — the bead you choose to offset from will suffice to identify the curve on which the relative Arc-length Bead will lie.

Location is the signed arc-length distance measured along Curve from t=0. If the Bead is relative to another bead, then the distance is as measured from that bead. Arc-length distance on Curve increases in the same direction as the parameter t.

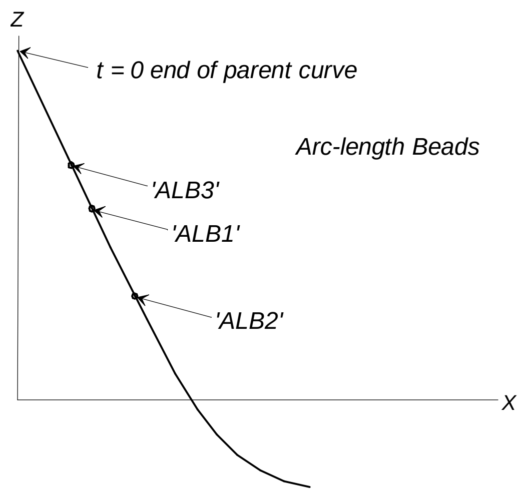
Relative to: If you want to make the Arc-length Bead relative to (offset from) another bead (or from a ring), check the box and specify the parent bead (or ring). (Be aware that if you pick a bead on another curve, the bead will jump to that curve). If you change the parent bead, or if you change between absolute / relative coordinates, SurfaceWorks asks if you want to "Preserve Absolute Location" or "Preserve Offset Value" of the Arc-length Bead.

The default location for an Arc-length Bead depends on whether it has absolute or relative coordinates:

When it has absolute coordinates, the default location of the new Arc-length Bead is at arc-length = 0, which is the t = 0 end of the parent curve.

When it has relative coordinates ("Relative to" is checked), the default location of the new Arc-length Bead is at arc-length offset = 0, which puts it at the location of the "Relative to" bead (or ring).

Example: ArcLengthBead



ArcLengthBead

The parent curve is a B-spline Curve; there are 3 Arc-length Beads:

'ALB1' has Curve = the curve and Location = 1.8, therefore it is located 1.8 units from the t = 0 end of the curve.

'ALB2' has Relative to = the bead 'ALB1' and offset Location = 1.0, therefore it is located 1.0 units from 'ALB1' in the direction of <u>increasing</u> t.

'ALB3' has Relative to = the bead 'ALB1' and its offset Location = -0.5, therefore it is located 0.5 units from 'ALB1' in the direction of <u>decreasing</u> t.

See also: Arc-length Ring, Bead, Ring, XYZ Bead, XYZ Ring

Arc-length Ring

Parents and characteristic properties

Snake = parent snake
arc-length (or arc-length offset when Relative to is checked) = arc-length distance along snake measured in the units of the model
Relative to (when checked) = the parent ring for relative location; also designates parent snake

Description

An Arc-length Ring is a ring positioned along a snake by arc-length distance (measured in the units of the model as opposed to parameter-t distance or X, Y, or Z coordinate). An Arc-length Ring is specified by naming the parent Snake and giving the arc-length distance along that snake (or arc-length distance offset from another ring on the same snake). An Arc-length Ring can be used wherever a point, bead, magnet, or ring is called for.

Snake can be any kind of snake. When Relative to is checked, the Snake field will gray out — the ring you choose to offset from will suffice to identify the snake on which the relative Arc-length Ring will lie.

Location is the signed arc-length distance measured along Snake from t=0. If the Ring is relative to another ring, then the distance is as measured from that ring. Arc-length distance on Snake increases in the same direction as the parameter t.

Relative to: If you want to make the Arc-length Ring relative to (offset from) another ring, which must be <u>on the same snake</u>, check the box and specify the parent ring. If you change the parent ring, or if you change between absolute / relative coordinates, SurfaceWorks asks if you want to "Preserve Absolute Location" or "Preserve Offset Value" of the Ring.

The default location for an Arc-length Ring depends on whether it has absolute or relative coordinates:

> When it has absolute coordinates, the default location of the new Arc-length Ring is at arc-length = 0, which is the t = 0 end of the parent snake.
>
> When it has relative coordinates (“Relative to” is checked), the default location of the new Arc-length Ring is at arc-length offset = 0, which puts it at the location of the “Relative to” ring.

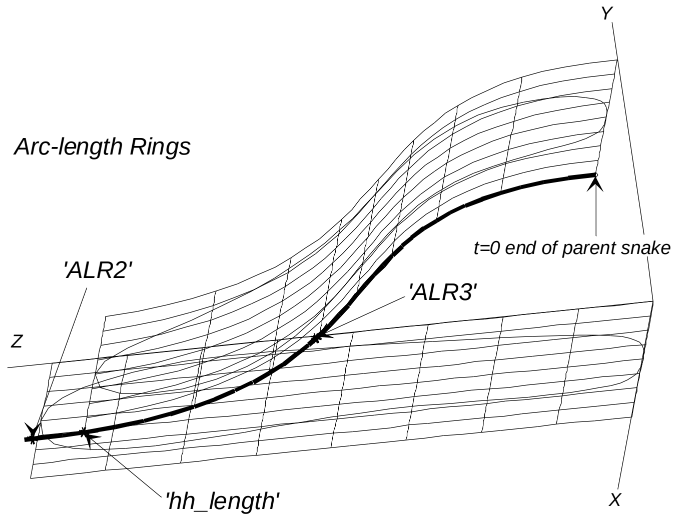
Example: CopySnake2-ArcLengthRing

In this model, the Arc-length Ring ‘hh_length’ is used to make the arc length of the curved surface for the high-heeled sole the same length as the flat surface with the flat sole drawn on it. The parent snake for ‘hh_length’ is a B-spline Snake ‘profile_full’ (magenta). The SubSnake ‘profile_sub’ (green), between ‘ring11’ and ‘hh_length’, forms the long edge of the curved surface. The other two Arc-length Rings are for demonstration purposes only — they play no useful role in the model.

‘hh_length’ has Snake = 'profile_full' and Location = 7.0, therefore it is located 7.0 inches from the t = 0 end of the snake.

‘ALR2’ has Relative to = the ring ‘hh_length’ and offset Location = .6, therefore it is located .6 inches from ‘hh_length’ in the direction of <u>increasing</u> t.

‘ALR3’ has Relative to = the ring ‘hh_length’ and offset Location = -3.0, therefore it is located 3.0 inches from ‘hh_length’ in the direction of decreasing t.



CopySnake2-ArcLengthRing

See also: Arc-length Bead, Bead, Ring, XYZ Bead, XYZ Ring, arclnrg.ms2

Arc Lofted Surface

Parents and characteristic properties

Type = one of 6 types of arc
Relabel = Relabels surface in lofting direction
Control curves = point entity or curve entity. curve1, curve2, curve3

Description

In the Arc Lofted Surface, the lofting curves are circular arcs. There are 6 types:

- arc through 3 curves — the arc passes through points taken from the 3 control curves

- arc: start-center-end guide — the arc starts at the point on curve1, takes its center from the point on curve2, and ends at the correct radius on a line from the center through the point on curve3

- circle: start-center-plane guide — the arc is a complete circle starting at the point on curve1, taking its center from the point on curve2, and lying in the plane of the 3 points; curve3 is used to set the direction of the circle — the closest point on the circle to the point taken from curve3 is at a parameter value t that is less than 0.5

- arc: tangent at curve1 end — the arc starts at the point on curve1, uses the point on curve2 for the middle point of its polyline, ends at the point on curve3, and is tangent to its polyline at the t=0 end

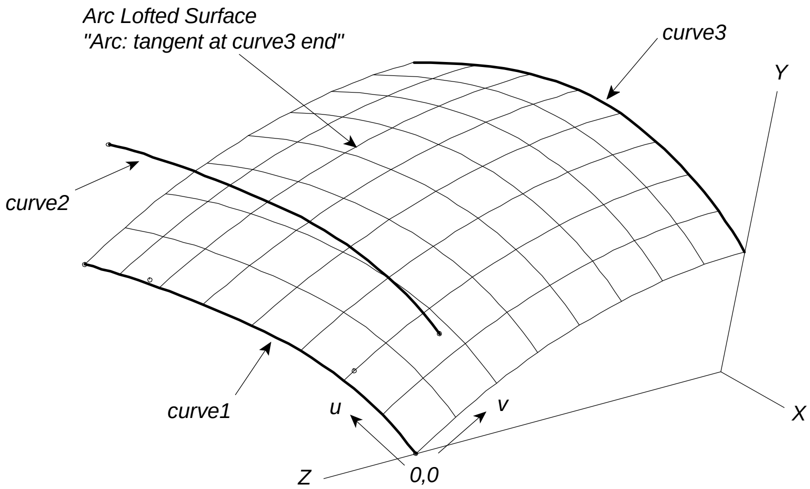
- arc: tangent at curve3 end — the arc starts at the point on curve1, uses the point on curve2 for the middle point of its polyline, ends at the point on curve3, and is tangent to its polyline at the t=1 end

- semi-circle: start-center-plane guide — the arc is a semi-circle that starts at the point on curve1, uses the point on curve2 as its center, and lies in the plane determined by the points on curve1, curve2, curve3; the point on curve3 determines which of the two possible semi-circles you get — the one that passes closest to that point

For details about how lofted surfaces are generated, see “Understanding SurfaceWorks Entities - Collective Entity Information - Lofted Surfaces”.

Relabel is used to control the labeling of the v-direction (the lofting curves) of the surface.

Example: ArcLoft1



ArcLoft1 Type = Arc: tangent at curve3 end

This surface has a non-planar curve as one edge. We created ‘surface1’ using an Arc Lofted Surface of Type “Arc: tangent at curve3 end”.

B-Spline Curve ‘curve1’ (cyan) is the only curve needed to be constructed from points to make this surface. It is the non-planar curve that defines one edge of the surface.

‘curve2’ (cyan) is a Relative Curve based on ‘curve1’.

‘curve3’ (cyan) is a projection of ‘curve2’ onto the *Z=0 centerplane.

Each circular arc that forms the surface ‘surface1’ uses 3 points of equal t-value, one from each of the 3 curves. Each arc starts on the point at ‘curve1’, takes the middle point of its polyline from ‘curve2’, and ends at the point on ‘curve3’. The lofting curves are tangent to their polylines at the t=1, or ‘curve3’, end. And since we made ‘curve3’ a projection of ‘curve2’ onto the centerplane, the polylines are perpendicular to the centerplane and a Mirrored Surface would join smoothly across the centerplane. If you use symmetry images (press <**F5**>), you can see that this is true.

See also: B-spline Lofted Surface, C-spline Lofted Surface, ArcLoft4.ms2

Arc Snake

Parents and characteristic properties

magnet1, magnet2, magnet3 = magnet or ring control points
type = 1 – 6

Description

An Arc Snake is a continuous curve lying in a surface and defined by 3 magnets, which must all lie in the same surface. In the u-v parameter space, the Arc Snake is a circular arc. There are 6 types:

type-1 arc (through 3 magnets) — the arc passes through all three magnets

type-2 arc (start-center-guide) — starts at magnet1, uses magnet2 for its center, and ends (at the correct radius) on a line from the center through magnet3

type-3 circle — the arc is a complete circle starting at magnet1 and using magnet2 for its center; magnet3 has no role except to set the direction of the circle; magnet3 is used to set the direction of the circle — the closest point on the circle to magnet3 is at a parameter value t that is less than 0.5

type-4 arc (tangent at start) — the arc starts at magnet1, uses magnet2 for the middle point of its polygon, ends at magnet3, and is tangent to its polygon at the t=0 end

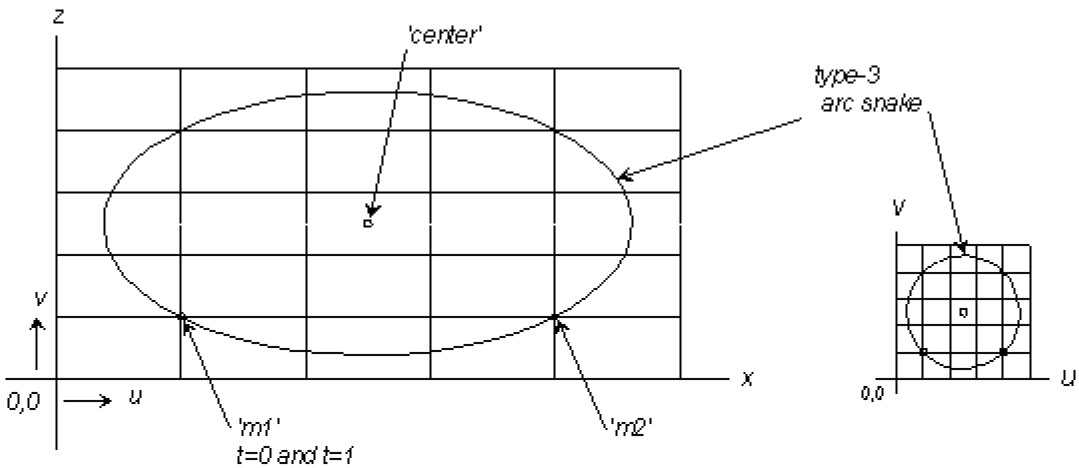
type-5 arc (tangent at end) — the arc starts at magnet1, uses magnet2 for the middle point of its polygon, ends at magnet3, and is tangent to its polygon at the t=1 end

type-6 semi-circle — the arc is a semi-circle starting at magnet1 and using magnet2 as its center. magnet3 determines which of the two possible semi-circles you get — the one that passes closest to magnet3

The program’s default relabel produces the “natural” labeling, in which the parameter t is uniformly distributed with respect to arc length or angle, in the u-v parameter space.

Example: ArcSnake

A type-3 arc snake, is shaped by the three magnets ‘m1’, ‘m2’, and ‘center’. The Arc Snake is a circle in the u-v parameter space mapped onto a rectangle ‘patch’. This produces a true ellipse. If the shape of ‘patch’ is changed, the snake will change its position in space but will remain always on ‘patch’, in a corresponding position. The Arc Snake will remain an ellipse as long as ‘patch’ remains a rectangle.



ARCSNAKE.MS2 in <Y> view.

See also: Arc

Bead

Parents and characteristic properties

Curve = parent curve for Bead
t (or dt when Relative to is checked) = location along curve
Relative to (when checked) = the parent bead for relative location

Description

A bead is a point constrained to stay on a curve. The Bead entity is specified by naming the Curve and a parameter value t along that curve (or dt offset from another bead on the same curve). The Bead entity can be used wherever a point or bead is called for.

Curve can be any kind of curve (including snakes). When Relative to is checked, the Curve field will gray out — the bead you choose to offset from will suffice to identify the curve on which the relative Bead will lie.

Relative to: If you want to make the Bead relative to (offset from) another bead (or from a ring), which must be on the same curve, check the box and specify the parent bead (or ring). If you change the parent bead, or if you change between absolute / relative coordinates, SurfaceWorks automatically will adjust the location value to maintain the Bead's position.

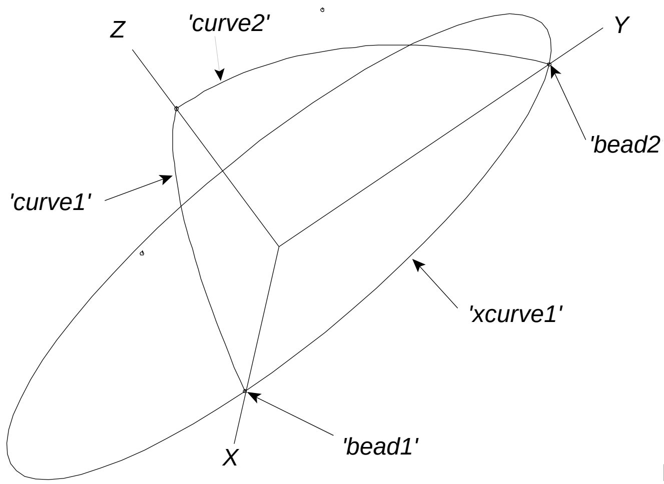
The default location for a Bead depends on whether it has absolute or relative coordinates:

When it has absolute coordinates, the default location of the new Bead is at t = 0.5 of the parent curve.

When it has relative coordinates ("Relative to" is checked), the default location of the new Bead is dt = 0, which puts it at the location of the "Relative to" bead (or ring).

Example: Beads1

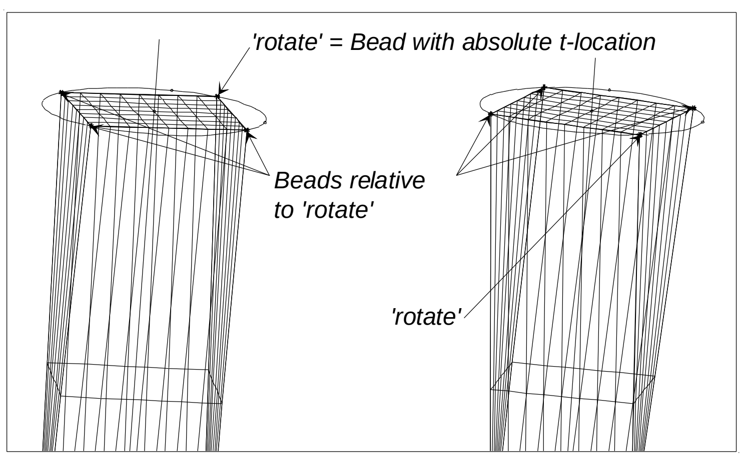
The Conic Curve which originated in another program and was imported ‘xcurve1’ (magenta) is an ellipse used to define the lower rim of a dome. Beads ‘bead1’ and ‘bead2’ (green) are located on ‘xcurve1’ at locations t=0 and t=.25, respectively. The beads are used as endpoints that durably anchor the quadrant profile curves of the dome, ‘curve1’ and ‘curve2’ (both cyan), to the rim.



Beads1

Example: Beads2

This example is a representation of the top of a sculpture on Detroit’s waterfront. We used four beads on an arc and created Lines between them to define the shape of the top of the tower. The Lines created between the beads are used in defining the tower surfaces.



Beads2: initial (left), with ‘rotate’ moved (right).

Bead ‘rotate’ (green) is a bead with absolute coordinates. It is the parent bead for the three relative Beads ‘relbead1’, ‘relbead2’, and ‘relbead3’ (all blue). The relative Beads will stay in the same relative position to the Bead ‘rotate’ when it is moved. Slide Bead ‘rotate’ anywhere along the circle to see the relative Beads move along with their child lines and surfaces.

See also: Ring, Arc-length Bead, Arc-length Ring, XYZ Bead, XYZ Ring

Blended Point

Parents and characteristic properties

Weight type
point1 ... pointN = component point names
weight1 … weightN = corresponding weights

Description

The Blended Point is a weighted sum of one or more points. Each coordinate of the blended point is the same weighted sum of the component point coordinates:

$X = w_1 X_1 + w_2 X_2 + ... + w_N X_N$

$Y = w_1 Y_1 + w_2 Y_2 + ... + w_N Y_N$

$Z = w_1 Z_1 + w_2 Z_2 + ... + w_N Z_N$

type specifies how the weight values are used in the formulas:

type = 0	weights are used without modification	$w_i = weight_i$
type = 1	weights sum to 1(forced)	$w_i = weight_i$ EXCEPT $weight_N$ is ignored -- it is replaced by 1 - (sum of the other weights)
type = 2	each weight is divided by the sum of all weights	$w_i = weight_i$ / sum

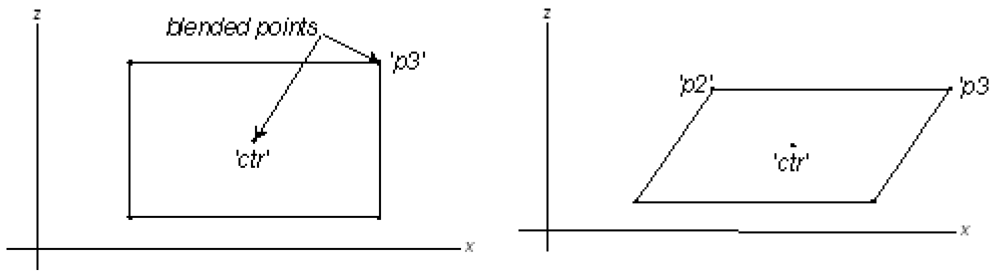
When weights sum to 1, it is generally easier to predict the result of the blend; when weights do not sum to one, the blend is more complex. For most applications, you probably want the weights to sum to 1. Thus, to construct the midpoint between two points, you could make a type-1 BlendPoint with weights of 0.5 and 0.5 . The weights used in the formulas would be $w_1 = 0.5$, $w_2 = 1 - 0.5 = 0.5$ (in a type-1 BlendPoint weightN is ignored and 1 minus the sum of the other weights is used). You could also use a type-1 BlendPoint to construct the mirror point of point1 across point2. You would use weights -1.0 and +2.0; the weights used in the formulas would then be $w_1 = -1.0$, $w_2 = 1 - (-1.0) = 2.0$.

Example: Blend Point2.ms2

Example 1: blendpt2

This model has two, type-1 (weights sum to 1) BlendPoints: ‘p3’ and ‘ctr’.

'p3' (cyan) is used to make the fourth corner of the parallelogram; its component points are ‘p0’, ‘p1’, and ‘p2’ (weights -1, +1, +1). Since ‘p3’ is a type-1 BlendPoint, the weights used in the formulas are: $w_1 = -1$, $w_2 = +1$, and $w_N = 1 - (-1 +1) = 1$. You can drag RelPoints ‘p1’ and ‘p2’ around as you wish, but the figure formed by BCurve 'c0' is durably parallel-sided.



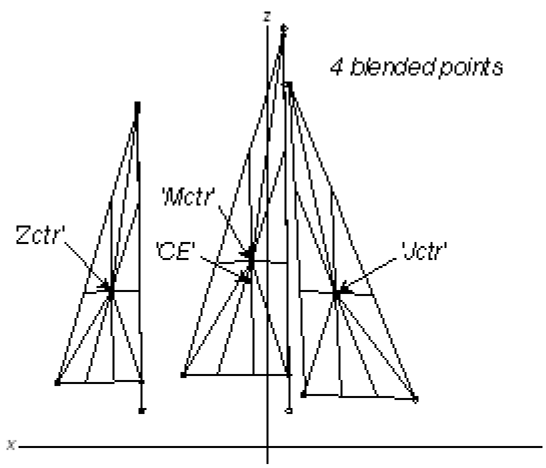
BLENDPT2.MS2 at Lat 0, Lon -90 orthographic: initial model (left); with ‘p2’ moved (right).

The second BlendPoint 'ctr' stays at the center of the figure; it is the average of the four corners ‘p0’, ‘p1’, ‘p2’, and ‘p3’ (weights .25, .25, .25, .25). Since ‘ctr’ is a type-1 BlendPoint, the weights used in the formulas are: w_1 = .25, w_2 = .25, w_3 = .25, and w_N = 1 - (.25 + .25 + .25) = .25 .

Example: Blend Point3.ms2

Example 2: blendpt3.ms2

This example uses three type-1 BlendPoints and one type-2 BlendPoint to make the standard yacht designer’s “center of effort” calculation.



BLENDPT3.MS2 at Lat 0, Lon 90 orthographic.

The “ketch” rig has three triangular sails: ‘jib’, ‘main’ and ‘mizzen’. Each sail is made from its 3 corner points. The center of each sail is located by a type-1 BlendPoint, equally weighting the 3 corners. (This is a valid way to locate the centroid of a triangle.) The boundary of each sail is a type-1 BCurve, and the surface of the sail is a RuledSurf between the centroid and the boundary. The “Center of Effort” (CE, the supposed aerodynamic center of sail force) is defined as the weighted sum of the individual sail centroids, each weighted by the area of that sail. The sail areas are obtained from Calcs / Mass Properties:

Sail	**Area**	**Area/sum of areas**
	(weights for the BlendPoint)	(w_1, w_2, w_3 values used in the formulas)
‘Jib’	275 sq.ft.	.3681
‘Main’	288 sq.ft.	.3855

'Mizzen'	184 sq.ft.	.2463
Total	747 sq.ft.	

A type-2 BlendPoint 'CE' makes the center of effort calculation; the three sail areas are entered as the weights. Since 'CE' is type-2, the weights used in the formulas are w_1 = .3681, w_2 = .3855, and w_3 = .2463 (as shown in the table above). You can read the absolute coordinates of the center of effort 'CE' from the Edit / Attributes box or the status line.

[Variables and Formulas provide a way to represent this complete relationship durably. The critical element is the AREA function.]

See also: Ring, Arc-length Bead, Arc-length Ring, XYZ Bead, XYZ Ring

Blend Surface

Parents and characteristic properties

Surface blend continuity = G1 or G2
Snake 1= snake which identifies the outer boundary of the Blend Surface
Snake 2 = snake that helps the merging of the Blend Surface into its parent surface
Snake 3 = snake that helps the merging of the Blend Surface into its parent surface
Snake 4= snake which identifies the outer boundary of the Blend Surface

Description

A Blend Surface provides a smooth local blending between two basis surfaces (or between two ends of a single surface). It is defined by four snakes — two on each of the basis surfaces (or all four on the single basis surface).

Blend boundary on each of the surfaces specifies the outer boundary of the blend on that surface. Blend helper on each of the surfaces is a snake that guides the interior shape of the Blend Surface — it gives the direction for the Blend Surface to leave the basis surface. Blend boundary and Blend helper for Surface 1 must both lie on Surface 1; Blend boundary and Blend helper for Surface 2 must both lie on Surface 2.

The distribution of t along the four snakes should be relatively similar — otherwise the Blend Surface may have unwanted folds or waves or may gouge into the basis surfaces. You can visualize the t-distributions by turning on tickmarks for each snake (on the Edit>Definition>Display tab). You can use SubSnakes to adjust the t-distribution of snakes as necessary.

The u parameter on the Blend Surface has the same orientation as the t parameter of the Surface 1 Blend boundary snake. The v = 0 edge of the Blend Surface is along that Blend boundary snake, and the v = 1 edge is along the Blend boundary snake of Surface 2.

Surface blend continuity controls the smoothness of the join between the Blend Surface and the basis surfaces, as well as the radius of the blend:

G1: The Blend Surface agrees with the surfaces in slope

G2: Makes a smoother join to the two surfaces than a G1 blend — the Blend Surface agrees with the surfaces in curvature as well as slope; also makes a tighter blend — radius about 2 / 3 and cross-sectional area only about 1 / 2 as big as a G1 blend.

Relabel is used to control the labeling of the v-direction (the lofting curves) of the surface.

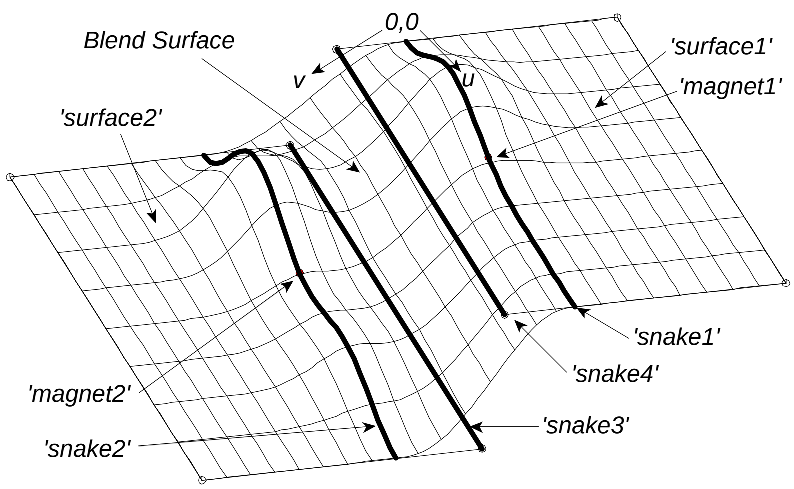
Example 1: BlendSurface1

This model uses a Blend Surface to unite two Tangent Boundary Surfaces, blending one into another with G2 continuity.

‘surface1’ and ‘surface2’ (both gray) are the two Tangent Boundary Surfaces. ‘surface3’ (red) is the Blend Surface joining the two Tangent Boundary Surfaces. Its snake parents (all magenta) are:

Snake 1 on Surface 1 = ‘snake1’ (a UVSnake)
Snake 2 on Surface 1 = ‘snake4’ (an Edge Snake)
Snake 3 on Surface 2 = ‘snake3’ (an Edge Snake)
Snake 4 on Surface 2 = ‘snake2’ (a UVSnake)

Since they are UVSnakes, the Blend boundaries ‘snake1’ and ‘snake2’ can be moved further toward or away from their basis surface edge by dragging ‘magnet1’ and ‘magnet2’ (red) respectively. And of course, you can modify the shapes of the Tangent Boundary Surfaces any way you like and the Blend Surface will still connect them smoothly.



BlendSurface1

The surfaces you want to use in your finished model are the Blend Surface (red) and two SubSurfaces (both green) which are built between the Blend boundaries and the outer edge of their respective basis surfaces.

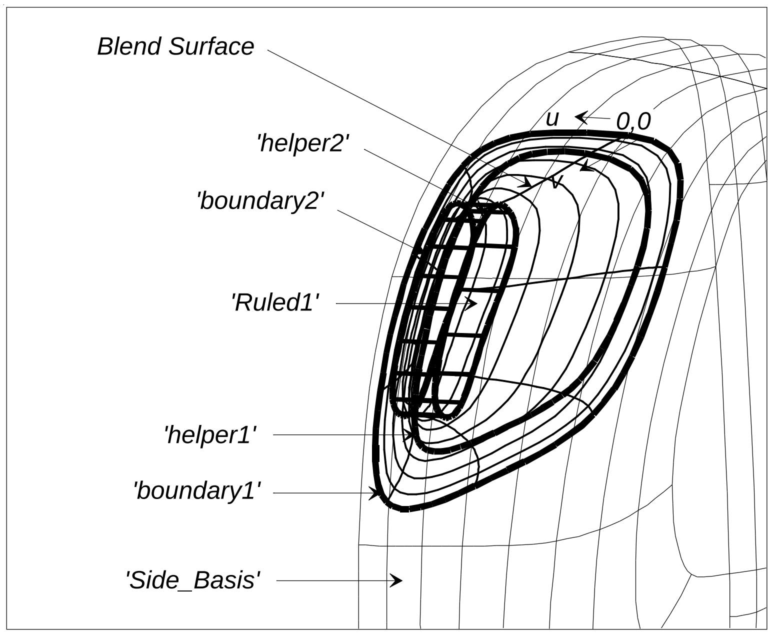
Example 2: BlendSurface2

This model is like the model in the Bottle Handle tutorial, except that the handle is made with a Blend Surface rather than a B-spline Lofted Surface, to ensure G2 continuity both between the side of the bottle and the handle, and across the centerline of the handle itself.

‘Blend1’ (green) is the Blended Surface that forms the handle. Its four parent snakes are ‘boundary1, ‘helper1’, ‘helper2’ and ‘boundary2’. They all have t-orientation running in the same direction, beginning at the “top center” of the loop. Here’s how the snakes were constructed:

‘Side_Basis’ (gray; a Degree-2 B-spline Lofted Surface) is the basis surface for the side of the bottle. Among other roles, it serves as Surface 1 for the Blend Surface. ‘boundary1’ and ‘helper1’ (both magenta) are B-spline Snakes that lie on it.

‘Ruled1’ (blue; a Degree-2 B-spline Lofted Surface) was created specifically to make the handle halves meet normal to the bottle’s centerplane. It serves as Surface 2 for the Blend Surface. Its parents are ‘curve3’, a curve in the centerplane of the bottle, and ‘curve4’, a Relative Curve off ‘curve3’, offset normal to the centerplane (both of these are hidden). The Edge Snakes ‘boundary2’ and ‘helper2’ (both magenta) lie on ‘Ruled1’.



BlendSurface2

See also: Blister, Tangent Boundary Surface

Blister

Parents and characteristic properties

Type = 0 - 15
Snake1 = snake
Snake2 = snake
Apex = point entity or curve entity

Description

A Blister surface is a protuberance or bulge, usually a local feature, built up on a parent surface. Snake1 identifies the parent surface and specifies the boundary of the blister on that surface. Snake2 is a snake, magnet, or ring on the same surface as Snake1, generally located inside Snake1. Apex is a curve or point entity that specifies the apex of the blister.

Style selects the characteristic profile of the blister and the way it contacts the parent surface. There are 16 styles (Fig. 1):

style-0 rectangular profile; floats above the surface

style-1 elliptical profile; contact is approximately perpendicular to the parent surface

style-2 parabolic profile; contact has slope discontinuity

style-3 doubly-inflected profile; contact with parent surface is slope continuous, with a small fillet

style-4 doubly-inflected profile; contact with parent surface is slope continuous, with a larger fillet

style-5 doubly-inflected profile; contact with parent surface is slope and curvature continuous, with a small fillet

style-6 doubly-inflected profile; contact with parent surface is slope and curvature continuous, with a larger fillet

style-7 just one more step in the trend established by styles 2-6

style-8 identical to style 0

styles 9-15 comprise a “second generation” of Blisters whose junction with the base surface along Snake1 is similar to styles 1-7 respectively, but which end at Apex with a slope rather than a rounded summit. When Apex is a point, the style 9-15 blisters come to a sharp (conical) point there, rather than a rounded dome (Fig. 1).

In Fig. 1, the sixteen styles of Blister are demonstrated with:

Snake2 = a magnet

Apex = a point

Many useful blister applications utilize a magnet for snake2, and a point for the apex. For styles 1 - 7, the top of the blister is rounded, and the tangent plane to the blister at the apex is parallel to the tangent plane to the parent surface at the magnet. Although the u=constant lines converge at the apex and there is a coordinate singularity there, the surface is actually smooth at the apex. Styles 9 - 15 have a conical shape as the point apex is approached.

The u parameter on a blister is the same as the t parameter on Snake1, Snake2, or Apex. The v=0 edge of the Blister surface is at Snake1; the v=1 edge is at Apex.

Relabel is used to control the labeling of the v-direction of the surface.

A point xyz at parameter values (u,v) is located by the following construction:

1 Snake1 and Snake2 are evaluated at t = u, resulting in two surface points s1 and s2

2 a line snake is constructed from s1 to s2, and evaluated at parameter v, resulting in surface point sp

3 Apex is evaluated at t = u, resulting in a space point c

4 xyz is located by moving from sp, along a vector equal to (c - sp) f(v), where f(v) is a scalar function depending on Style

Snake1 can be either an open or a closed curve. If it is a closed curve, the Blister surface will be shaped like a canopy, turret, or dome. If Snake1 is not closed, the Blister surface will be a portion of a blister or dome, resembling a scoop. Further, if snake1 is not closed AND if Snake2 is a magnet located along the line snake connecting the two ends of Snake1, the u=0 and u=1 edges of the scoop will meet with continuous slope at the apex.

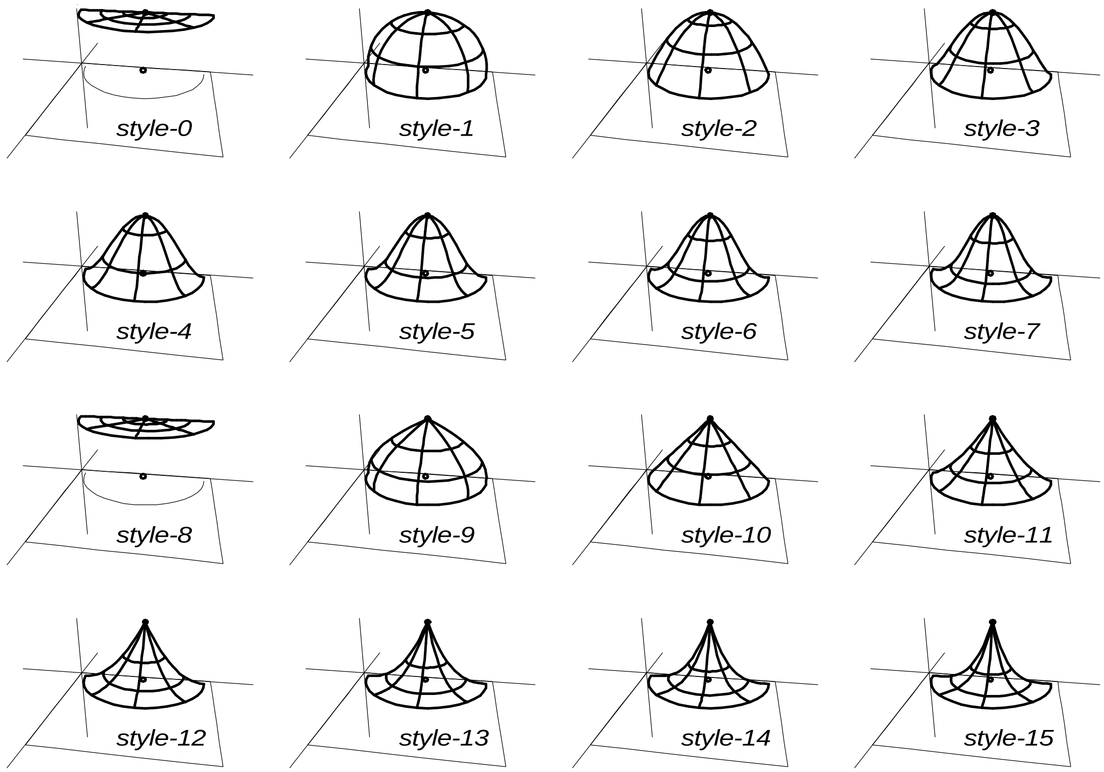


Fig. 1. Sixteen styles of Blister surface, demonstrated with:
Snake2 = a magnet
Apex = a point

Many useful blister applications utilize a magnet for Snake2, and a point for the Apex. For styles 1 - 7, the top of the blister is rounded, and the tangent plane to the blister at the apex is parallel to the tangent plane to the parent surface at the magnet. Although the u=constant lines converge at the apex and there is a coordinate singularity there, the surface is actually smooth at the apex. Styles 9 - 15 have a conical shape as the point apex is approached.

Example: Blister0

This is an example of a Blister surface using point entities for Snake2 ('center') and Apex ('apex').

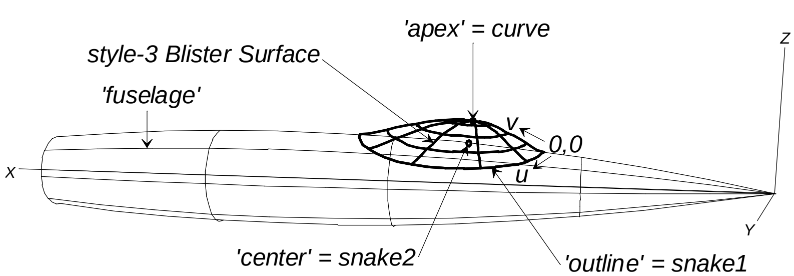


Fig. 2. Blister0

'fuselage', the right half of the fuselage, is a Revolution Surface. 'fuselage_mirror' is the mirrored left half of the fuselage (hidden).

'outline' (magenta; obscured by the cyan blister surface) is a snake on 'fuselage' and 'center' (red) is a magnet (= Snake2), also on 'fuselage'.

These two entities plus the point 'apex' (black; = Curve) form 'canopy' (cyan), a style-3 blister. Since 'outline' is constrained to lie on 'fuselage', 'canopy' will always stay attached to 'fuselage', no matter how the shape of 'fuselage' may be changed.

Example: Blister1

This Blister example also uses point entities for Snake2 and Apex.

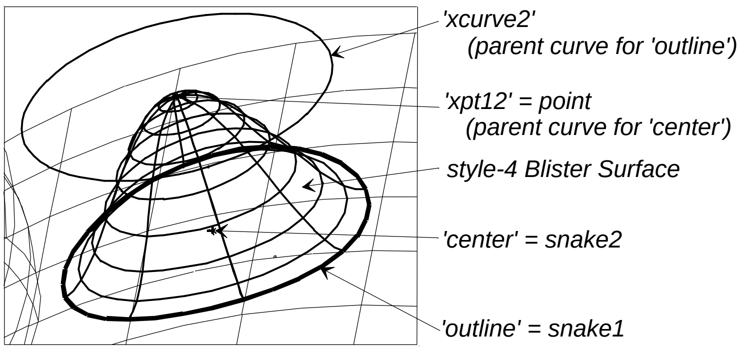


Fig. 3. Blister1

Translation Surface 'surface1' is first created using 'xline1' and 'xcurve1' as its parents. 'outline' (bright magenta; obscured by the blue blister surface) is formed by projecting the circular arc 'xcurve2' onto 'surface1'. 'outline' is then used as parent Snake1 for the style-4 Blister Surface (blue).

The Blister's other two parents are the point 'xpt12' (magenta; at the apex of the dome) and the Projected Magnet 'center' (red), a projection of 'xpt12' onto 'surface1'.

See also: Blend Surface, Tangent Boundary Surface

Block Solid

A Block Solid is a very simple solid requiring minimal parents. It is a rectangular solid, oriented by a frame and otherwise supported by just two points at two opposite corners.

Parents and characteristic properties

u-Relabel, v-Relabel, w-Relabel -- optional relabels for the 3 parametric directions
Frame
Point1
Point2

Description

The block is oriented parallel to the Frame coordinates. Its u-, v- and w-directions are along the Frame's x-, y- and z-axes respectively. Its u=0, v=0, w=0 corner is at Point1, and its u=1, v=1, w=1 corner is at Point2. The three Relabels apply to the three coordinate directions.

Sample files

BlockSolid2.ms2 is the simplest example of a block. It uses the default frame '*' , so its faces are parallel to the global coordinate system.

Boundary Solid

A Boundary Solid is parented by six surrounding surfaces, analogous to the way a Tangent Boundary Surface is parented by four surrounding curves. If the surfaces meet accurately along their junctions, the Boundary Solid will exactly fill the volume they bound. In the language of grid generation, Boundary Solid implements "transfinite interpolation" between its six bounding surfaces, although our implementation, using three relabels as blending functions, is more general than the standard method.

Parents and characteristic properties

u-graph
v-graph
w-graph
Bounding surfaces

Description

Exactly six bounding surfaces are required in all cases. Some of the surfaces can be degenerate, i.e., collapsed to a curve or a single point. This allows the Boundary Solid to have the topology of a prism (one face degenerated to a curve) or a pyramid (one face degenerated to a point), for example.

The u,v,w directions of the Boundary Solid depend entirely on which surface parent is first in the list of "Bounding surfaces". The first surface parent becomes the w = 0 face, and its u and v parameter directions become the u and v parameter directions for the solid. A non-degenerate surface should be chosen for the first parent, otherwise the orientation of the solid may be unstable, or it may not fill the volume enclosed by the other bounding surfaces.

Sample files

Boundary Solid1.ms2 is a simple rectangular block whose faces are 6 BSurfs.

Boundary Solid-Prism.ms2 is a prismatic block with a degenerate BSurf 's5' as one of its faces. If you move 's5' to the top of the list of parents, the resulting solid does not fill the volume enclosed by the surface parents.

Breakbead

Parents and characteristic properties

Curve = Parent Curve
Max Degree (-1 to …) = Maximum degree allowed
Index = which breakpoint on curve

Description

A bead located at a breakpoint of a curve.

Curve = Parent curve of Break Bead

Max Degree (-1 to …) = Maximum degree allowed
Index = Index number of breakpoint to which Breakpoint will be assigned.

Example: BreakBead_Ring.MS2

Breakring

Parents and characteristic properties

Snake = Parent Snake
Max Degree (-1 to …) = Maximum degree allowed
Index = which breakpoint on snake

Description

A ring located at a breakpoint of a snake.

Snake = Parent curve of Break Bead
Max Degree (-1 to …) = Maximum degree allowed
Index = Index number of breakpoint to which Break Ring will be assigned.

Example: BreakBead_Ring.MS2

B-spline Curve

Parents and characteristic properties

Degree = B-spline degree: 1 = linear, 2 = quadratic, 3 = cubic, etc.
Control points = point entity. point1 . . . pointN

Description

A B-spline Curve is a continuous curve defined by a series of control points. The curve is formed in relation to the 3D polyline (i.e., the light gray line) joining the points in sequence. The B-spline Curve always starts at the first control point (point1) and ends at the last control point (pointN), and it is always tangent to the polyline at these end points, but in general it does not pass through the other control points (unless the curve is degree-1).

degree-1 the curve is polyline itself (Fig. 1, left); this is one way to make a polyline entity in SurfaceWorks

degree-2 the curve is made of one or more parabolic arcs that join with continuous slope (except at doubled control points) (Fig. 1, middle)

degree-3 the curve is made of one or more parametric cubic segments that join smoothly with continuous slope and curvature (except at multiple control points) (Fig. 1, Right)

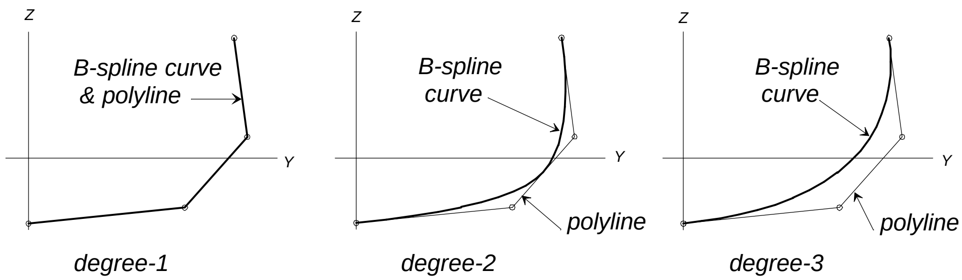


Fig. 1 Three degrees of B-spline Curve

For details about degrees of B-spline and how they behave, see "Understanding SurfaceWorks Entities - Collective Entity Information - B-spline Entities - Degrees".

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is concentrated in highly curved areas formed by closely spaced control points.

Example: BCurve

'c1' (cyan) is a degree-2 B-spline Curve created from four points. It lies in the X = 0 plane, since all of its points do. Both the B-spline Curve (cyan) and its polyline are displayed.

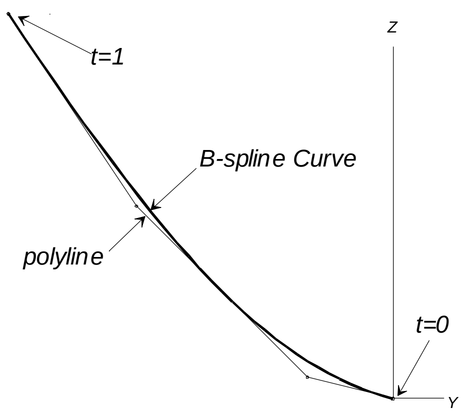


Fig. 2. BCurve.

See also: B-spline Lofted Surface, C-spline Curve, BSPL1-3.ms2

B-spline Fitted Curve

Parents and characteristic properties

Degree = B-spline type
No. of control pts = integer; number of control points to use for fit (0 if free)
Log-tolerance = log (base 10) of tolerance
Curve = basis curve

Description

A B-spline Fitted Curve is a uniform B-spline curve, least-squares fitted to the tabulated data of its basis curve. Its principal function is for exporting models to NURBS-based systems. Creating B-spline Fitted Curves before you File>Export 3D>IGES gives you more control over the accuracy of the exported model:

- you can see the NURBS approximation results right in SurfaceWorks, prior to exporting IGES
- you can specify type and number of control points for NURBS curve approximation
- you can specify different tolerances for different curves

Log-tolerance is the tolerance specified with a base-10 logarithm, e.g. -3 for tolerance of 10^{-3}. This notation permits a wide dynamic range. (Tolerance itself is measured as the RMS (root-mean-squared) deviation of the fitted curve from tabulated points of the basis curve.)

Degree is the type of B-spline : 1 = linear, 2 = quadratic, 3 = cubic, etc.

No. of control pts is an integer value which specifies the number of control points to be used. There are two choices:

When you specify a positive value (it must be greater than the corresponding Degree), the program will use exactly this number of control points for the fit.

When you specify zero, the program will start at Degree +1 control points and cycle No. of control pts upward until it either achieves the fit within the tolerance or reaches the limit of 128 control points.

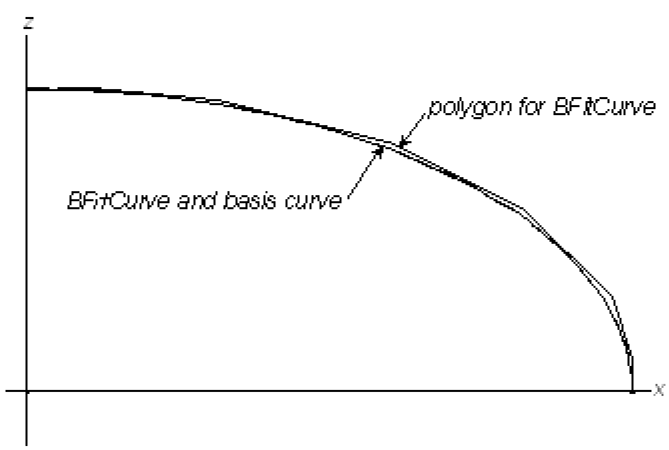
If the B-spline Fitted Curve does not meet the specified tolerance, an Error will be generated to let you know there is a problem.

When the B-spline Fitted Curve is displayed, you can see where the control points are by setting 'Show polyline' to true in the Entities Manager for the curve. The control points are located at the polyline vertices.

If you would like SurfaceWorks to report information about the quality of fit, select the B-spline Fitted Curve and then choose Tools>Measure>Clearance.

Example: BFitCv1.ms2

This example is a cubic (type-3) B-spline curve fitted to one quadrant of an ellipse basis curve. Since No. of control pts is 0, the program incremented the number of control points until the tolerance (1E-4 = .0001) was met. The fit was accomplished with 8 control points.



The deviation of the fitted curve from the basis curve, as provided by Tools>Measure>Clearance, is: rms = 6.80e-005.

See also: B-spline Fitted Snake, B-spline Fitted Surface

B-spline Fitted Snake

The B-spline Fitted Snake entity is very similar to the B-spline Fittted Curve. Its parent must be a snake, and the resulting approximation is in the u,v, space of the supporting surface.

Its control points are u,v locations on the supporting surface. The “tolerance” in logtolerance is a dimensionless value in u,v-space, rather than a distance.

See also: B-spline Fitted Curve, B-spline Fitted Surface

B-spline Fitted Surface

Parents and characteristic properties

u-Degree, v-Degree = B-spline degrees for u and v directions: 1, 2, 3, etc.
No. of u control pts, No. of v control pts = integers; number of control points to use for fit (0 if free)
Log-tolerance = log (base 10) of tolerance
Surface = parent surface

Description

A B-spline Fitted Surface is a uniform B-spline surface, least-squares fitted to the tabulated data of its parent surface. Its principal function is for exporting, via IGES, surfaces that do not automatically meet the tolerance when exported. In such cases, creating a B-spline Fitted Surface from the parent surface for IGES transfer gives you more control over the accuracy of the exported surfaces:

- you can see the approximation results right in SurfaceWorks, prior to export
- you can assure accurate joins between surfaces (by forcing the same number of control points along adjoining surface edges)
- you can specify different tolerances for different surfaces

u-Degree and v-Degree are the spline type for the u and v directions respectively: 1 = linear, 2 = quadratic, 3 = cubic, etc.

No. of u control pts and No. of v control pts are integer values that specify the number of control points to be used in the u- and v-directions. For each specifier there are two choices:

> When you specify a positive value (it must be greater than the corresponding degree), SurfaceWorks will use exactly this number of control points for the fit.
>
> When you specify zero, SurfaceWorks will start at degree+1 control points and cycle the number upward until it either achieves the fit within the tolerance or reaches the limit of 128 control points.

Log-tolerance is the tolerance specified with a base-10 logarithm, e.g. -3 for tolerance of 10^{-3}. This notation permits a wide dynamic range. (Tolerance itself is measured as the RMS (root-mean-squared) deviation of the fitted surface from tabulated points of the parent surface.)

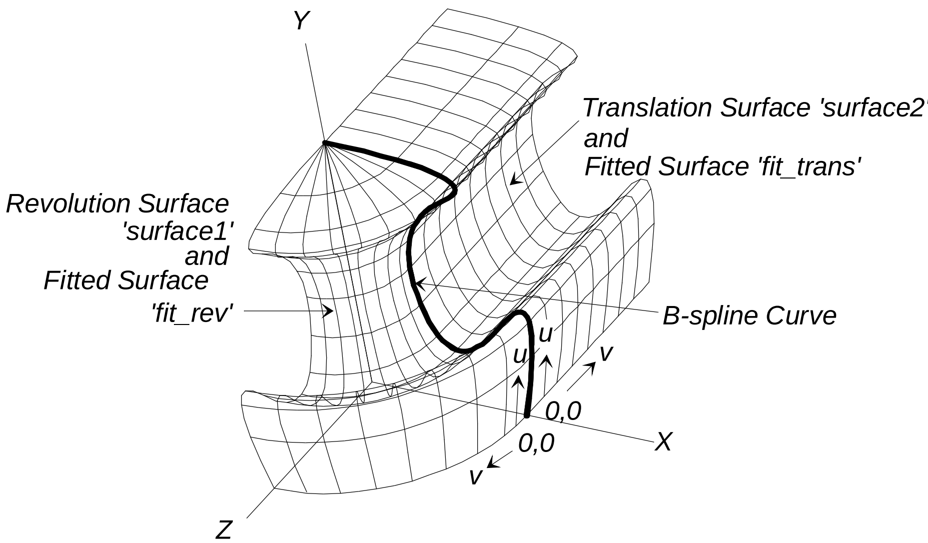
If the B-spline Fitted Surface does not meet the specified tolerance, an Error will be generated to let you know there is a problem.

When the B-spline Fitted Surface is displayed, you can see where the control points are by setting 'Show net' to 'true' in the Entities Manager for the surface. The control points are located where the net lines meet.

If you would like SurfaceWorks to report information about the quality of fit, select the B-spline Fitted Surface and then choose **Tools>Measure>Clearance**.

Example: FittedSurfs

In this model, the initial surfaces are a Revolution Surface ‘surface1’ and a Translation Surface ‘surface2’ (both green). As we wanted to make sure ahead of time that our SurfaceWorks surfaces would be accurately represented as NURBS in the IGES file, we used B-spline Fitted Surfaces to generate the NURBS and then viewed them in SurfaceWorks to check for problems. The fitted surfaces exported quickly and without problem.



FittedSurfs.

For ‘fit_rev’ (blue) we used: u-Degree = 2, No. of u control pts = 6 (to exactly match the degree-2 B-spline control curve built from six control points), v-Degree = 3, No. of v control pts = 4, and Log-tolerance = 0.

For ‘fit_trans’ (also blue) we used: u-Degree = 2, No. of u control pts = 6 (same as for the adjacent edge of ‘fit_rev’), v-Degree = 1, No. of v control pts = 2 (in this direction, the Translation Surface is based on a line), and Log-tolerance = 0.

By specifying the same u-Degree and No. of u control pts for each of the B-spline Fitted Surfaces, we ensured accurate stitching along their common edge.

See also: B-spline Fitted Curve, B-spline Fitted Snake, BFitSurface.ms2

B-spline Graph

Parents and characteristic properties

Degree = type of B-spline used in graph: 1 = linear, 2 = quadratic, 3 = cubic, etc.

value1 ... valueN = values for graph control points

Description

A B-spline Graph is a graph of some function f vs. t from 0 to 1, represented as a sum of B-splines (with uniform knots):

$$f(t) = \sum_{j=1}^{N} f_j B_j(t) \qquad (1)$$

type is a positive integer specifying the B-spline type used in the B-spline Graph formulas: 1 = linear, 2 = quadratic, 3 = cubic, etc. The higher the type, the stiffer the B-spline (for more information on B-splines, see "Understanding SurfaceWorks Entities - Collective Entity Information - B-spline Entities".

The braces enclose a list of values. As you may have noticed, the data format is the same as for a Relabel object, except the first and last components of the list are not forced to 0 and 1 — for instance, { -.20 -.02 0 } would be a perfectly legitimate values list for a B-spline Graph.

Like Knot List and Relabel, a B-spline Graph has no color or visibility, and it is not directly visible in the 3D modeling space, although its effects certainly may be visible.

B-spline Graphs are used as blending or distributing functions in a number of entities.

Example 1 TBS_Graph.MS2

This example uses B-spline Graphs to control the shape of a Tangent Boundary Surface.

Example 2 MastTaper.ms2

This example shows a graph being used to taper the mast to 65% of the maximum fore/aft thickness. Another reason to use a graph in this example is to hold the maximum thickness until approx 3/4 of the length and then taper from there.

See also: CenterPoint Boundary Surface, Tangent Boundary Surface, Offset Curve, Relative Curve, SweepSurf, RELCURV1.ms2

B-spline Lofted Solid

Parents and characteristic properties

Degree = type for the lofting B-splines; 1, 2, 3, etc.
Master surfaces = control surfaces

Description

The B-spline Lofted Solid is a solid between two or more supporting surfaces. It is generated by evaluating the point on each of the surfaces at each given u,v and constructing a B-spline curve from the points. The B-spline Lofted Solid interpolates its first surface (Master surface1) and last surface (Master surfaceN), but in general not the others, which have a "guiding" or "shaping" effect like the interior control points of a B-spline Curve

Technically, a B-spline Lofted Solid should have at least one more control surface than its Degree, e.g. at least 2 for type-1, 3 for type-2, etc; therefore, a B-spline Lofted Solid should have at least one more supporting surface than its Degree. However, SurfaceWorks will automatically “demote” a spline to lower type as necessary to fit the number of control points. For example, a type-3 B-spline Lofted Solid specified with only 2 surfaces will be treated as type-1; with 3 surfaces it will be treated as type-2; only with 4 or more surfaces will it actually be a cubic solid.

The supporting surfaces must have similar u,v orientation — the 0,0 corners need to be in a relatively similar place, and u and v must run in similar directions (otherwise the solid could wind up with an unexpected twist).

u and v for the solid run in the same directions as u and v for each supporting surface. w is the parameter along the lofting B-splines.

relabel is used to control the labeling of the w-direction (the lofting curves) of the solid. The default relabel ‘*’ produces the “natural” labeling, in which the parameter w is concentrated in highly curved areas formed by closely-spaced control surfaces.

Example: bloftsolid1.ms2

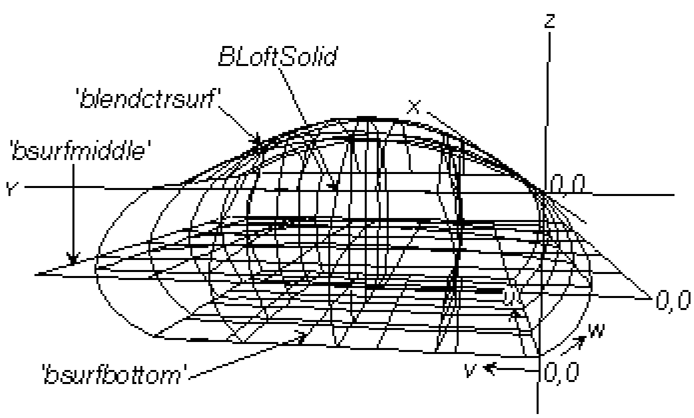
This B-spline Lofted Solid is made from three supporting surfaces:

surface1 = ‘blendctrsurf’ (cyan), a Centerpoint Boundary Surface

surface2 = ‘bsurfmiddle’ (green), a flat BSurf

surface3 = ‘bsurfbottom’ (magenta), another flat BSurf

Each of the lofting B-splines has three control points, taken from the point at the same u,v on each of the three surfaces. These lofting B-splines are type-2. As you can see, the B-spline Lofted Solid passes through its first and last supporting surfaces, but not its middle one. The w parameter for the solid runs along these B-splines; w = 0 is at the first supporting surface and w = 1 at the last one.



BLOFTSOLID1.MS2 at Lat 6, Lon -10.

B-spline Lofted Surface

Parents and characteristic properties

Relabel = Relabels surface in the lofting direction
Degree = longitudinal B-spline degree: 1, 2, 3, etc.
Master curves = point entity or curve entity. curve1 ... curveN

Description

In the B-spline Lofted Surface, the lofting curves are B-splines. This means the B-Lofted Surface passes through its start (curve1) and end (curveN) control curves, but in general not the others (unless the surface is degree -1), which have a "guiding" or "shaping" effect like the interior control points of a B-spline Curve. For details about B-spline degrees and behaviors, see "Understanding SurfaceWorks Entities - Collective Entity Information - B-spline Entities - Degrees". For details about how lofted surfaces are generated, see "Understanding SurfaceWorks Entities - Collective Entity Information - Lofted Surfaces"

Technically, a B-spline should have at least one more control entity than its degree, e.g. at least two for degree-1, three for degree-2, etc. However, SurfaceWorks will automatically "demote" a spline to lower degree as necessary to suit the number of control entities. For example, a degree-3 spline specified with only two entities will be treated as degree-1; with three entities it will be treated as degree-2; only with four or more entities will it actually be a cubic spline.

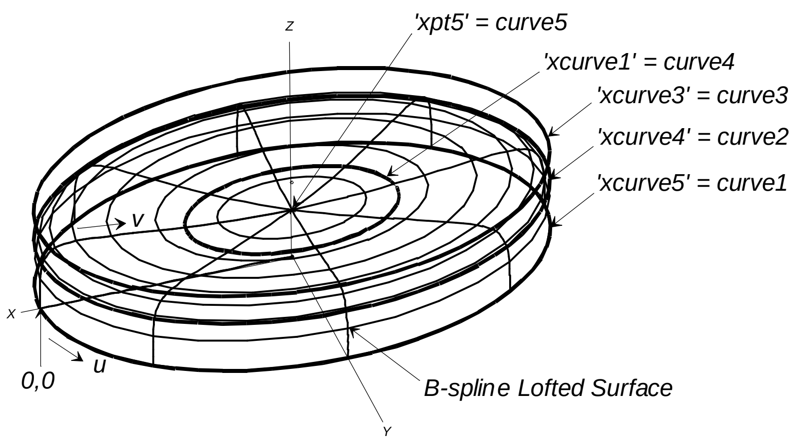
The u parameter on the B-spline Lofted Surface is the same as the t parameter for the first control curve. The v parameter is the parameter along the lofting B-splines, which strongly reflects the spacing of the control curves. The lofting B-splines are the u=constant lines.

A B-spline Lofted Surface with just two control curves makes a ruled surface, since a B-spline with two control points is a straight line.

Relabel is used to control the labeling of the v-direction (the lofting curves) of the surface.

Example: BLoftSurf

This B-spline Lofted Surface (green) is made from 5 imported parents: 'xcurve5', 'xcurve4', 'xcurve3', 'xcurve1' and point 'xpt5', selected in that order. The B-spline Lofted Surface passes through its first and last control curves ('xcurve5' and 'xpt5'), but not the others—those interior control curves only guide the shape of the surface. For comparison, see the C-spline Lofted Surface example, which uses the same set of control curves as parents, but has a surface that passes through all of the control curves. Notice also that this model is tangent across the center (a B-spline is tangent to its polyline at its endpoints), whereas the C-spline Lofted model has a little peak at its center.



BLoftSurf

See also: Arc Lofted Surface, C-spline Lofted Surface, B-spline Curve, BLFT5X4.ms2

B-spline Snake

Parents and characteristic properties

Degree = B-spline degree: 1 = linear, 2 = quadratic, 3 = cubic, etc.
Control magnets = magnet or ring. magnet1 . . . magnetN

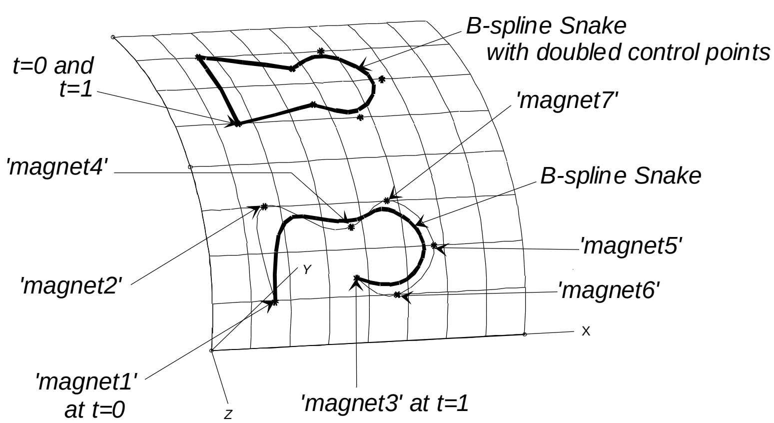
Description

A B-spline Snake is a continuous curve lying in a surface and defined by a series of magnets for control points. The magnets must all lie in the same surface. The curve ends at the two end control points (magnet1 and magnetN), but in general does not pass through the other control points (unless the snake is degree-1). Instead, it is “molded” or “shaped” by them into a continuous imitation of the polyline snake joining the control points in series. In the u-v parameter space, the B-spline Snake is a true B-spline Curve.

For details about B-spline degrees and how they behave, see “Understanding SurfaceWorks Entities - Collective Entity Information - B-spline Entities - Degrees”.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is concentrated in highly curved areas formed by closely spaced control points, in the u-v parameter space.

Example: BSnake



BSnake

The degree-2 B-spline Snake (magenta) shown in the lower part of the figure uses magnets ‘magnet1’ through ‘magnet7’ as its control points. The b-spline curve passes through only its endpoints; it is only guided by the others. The other curve in the lower pair (dark cyan), for comparison, is a degree-2 C-spline Snake using the same set of control points; it passes through all of its control points.

‘magnet1’ is a Magnet (with absolute u,v coordinates). All the other magnets are relative to 'magnet1' or each other. Magnets 2, 3, 5, and 6 are based on ‘magnet1’. Magnets 4, and 7 are based on magnets 3, and 6 respectively. The set of control points and the snakes can be moved anywhere on the surface simply by moving ‘magnet1’.

The degree-2 B-spline Snake (blue) uses a similar set of control points, ‘magnet1a’ through ‘magnet7a’. The angles are made by selecting the corresponding control point twice. Starting at ‘magnet1a’, ‘magnet2a’ was selected next but chosen twice to make the sharp angle. The selection goes on as follows: ‘magnet4a’ (twice), ‘magnet7a’, ‘magnet5a’, ‘magnet6a’, ‘magnet3a’ (twice), and back to ‘magnet1a’.

If the surface is modified in any way, the snakes will change their position in space, but will always remain on the surface in a corresponding position.

See also: B-spline Curve, C-spline Snake, BSNAK1-3.MS2

B-spline Solid

A B-spline solid (or BSolid) is constructed from control points in a way that is very parallel to BCurve and BSurf entity types, but extended to one more dimension.

Parents and characteristic properties

u-Relabel, v-Relabel, w-Relabel -- optional relabels for the 3 parametric directions
u-type, v-type, w-type -- B-spline types for the 3 parametric directions
ncu -- number of control points in the u direction
ncv -- number of control points in the v direction
Control points -- (ncu x ncv x ncw of them)

Note: BSolid has a "Net" visibility option, like the "Polygon" option for a BCurve, or the "Net" option f or a BSurf.

Sample models

BSolid1.ms2 is a minimal example of a BSolid, with just 8 control points. Its B-spline type is specified as 3 (cubic) in all 3 directions, but because there are only 2 control points in each of the 3 directions, all the B-splines are automatically demotes to degree-1 (linear).

B-spline Surface

Parents and characteristic properties

point1 ... pointMN = control points
utype, vtype = B-spline types for u and v directions: 1, 2, 3, etc.
ncu = no. of control pts. in u-direction
(ncv = in documentation below, no. of control points in v-direction)

Description

BSurf is a uniform B-spline surface. The control points form a quadrilateral **net** which guides or shapes the surface in much the same way as the control points shape a B-spline curve. In general, the surface does not interpolate any of its control points, except the four at the corners of the net. You can think of the surface as being connected to the interior control points by springs.

The edges are all B-spline curves using control points from the four edges of the net:

v = 0 uses the first ncu control points

v = 1 uses the last ncu control points

A B-spline Surface is a simplified case of a NURB Surface — the knots are uniform in both directions, and you don't need to use weights because they are all set to one. Technically, a B-spline should have at least one more control point than its type, e.g. at least 2 for type-1, 3 for type-2, etc. However, SurfaceWorks will automatically "demote" a spline to lower type as necessary to fit the number of control points. For example, a type-3 spline specified with only 2 points will be treated as type-1; with 3 points it will be treated as type-2; only with 4 or more points will it actually be a cubic spline.

Example: BSurf

Example: bsurf.ms2

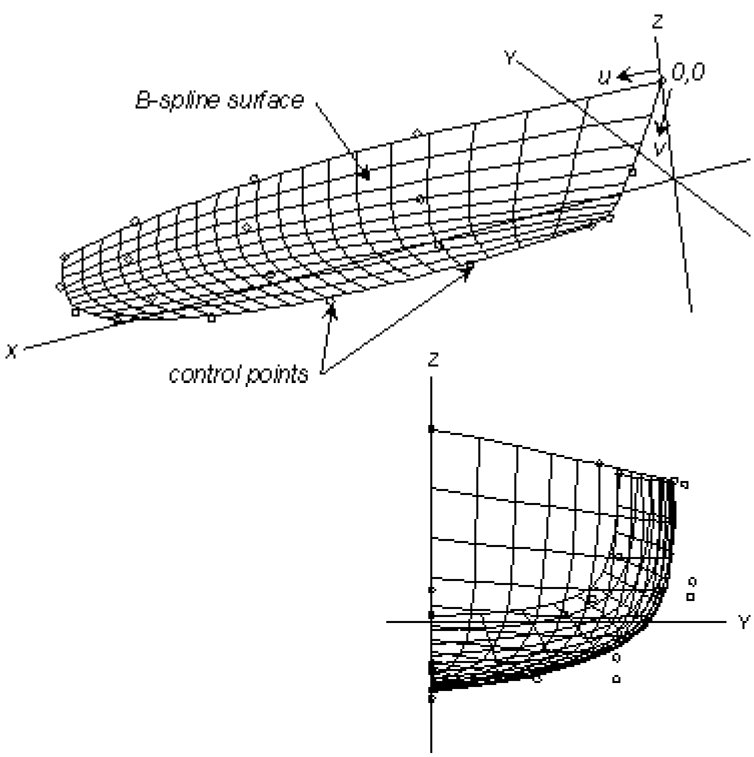
In this example, 'hull' is a B-spline surface which uses the same data points as the models:

CLFT5X4.MS2 in which 'hull_c' is a C-lofted surface ("CLoftSurf" example)

BLFT5X4.MS2 in which 'hull_b' is a B-lofted surface ("BLoftSurf" example 1)

Generally, a B-spline surface will lie inside a C-lofted surface defined by the same set of control points. A B-spline surface and a B-lofted surface defined by the same set of control points and using the same types of B-splines will be identical, but the BLoftSurf entity is much more flexible than the BSurf entity when it comes to refinement of surfaces.

For the surface 'hull' in this example, the u-direction (longitudinal, in this case) B-splines are type-3 (cubic), the v-direction B-splines are type-2 (quadratic), and the number of control points in the u-direction is 5.



BSURF.MS2 at Lat -20, Lon 50 (top) and in <x> view (bottom).

See Also: BLoftSurf, CLoftSurf, NURBSurf

Center Point

Parents and characteristic properties

Bead/curve = Parent bead or curve

Description

Bead/curve If the parent is a bead (or ring) it determines both the curve being measured, and the particular location along the curve where the curvature measurement is taken.

If the parent is a curve (or snake), the curvature measurement is taken at t = 0.5.

Center Point entity type locates the center of curvature of a curve or snake in 3 dimensions. More precisely, the Center Point locates itself at the center of the osculating circle, at a given point along a curve. The osculating circle is the circle that lies in the tangent plane of the curve, and has the same curvature as the curve does at that point. The distance of the Center Point from the curve is the radius of curvature.

Note: If the curve has low curvature (i.e., it is nearly straight) at the specified location, then the radius of curvature is large and the Center Point will be far away. This can greatly increase the size of the model bounding box, which can cause various viewing and geometry anomalies. The most common manifestation of a large bounding box is that when the model file is opened, the visible geometry is small and occupies only a small part of the screen, until a Zoom All (<F9>) is performed.

Example files

CenterPt0.ms2. This file has 4 examples of Center Points:

(Upper right quadrant) CP1 is made from bead1 on Arc c1. CP1 remains fixed as bead1 is moved, since an Arc is part of a circle, and has a fixed center point.

(Upper left quadrant) CP2 is made from bead2 on Helix c2. This is a zero-pitch helix, so it too is a portion of a circle, and CP2 stays put as bead2 is moved.

(Lower left quadrant) CP3 is made from bead3 on BCurve c3. Although c3 looks pretty much like a circular arc, it's actually a parabola with continuously varying curvature, from a radius of 4 at either end, down to 1.414 at t = 0.5.

(Lower right quadrant) CP4 is made from bead4 on Line l1. Of course the line is straight (infinite radius of curvature), so CP4 gets error 347 regardless of the position of bead4.

CenterPoint.MS2 – Measuring the distance between a bead on the deck parent and the center point shows the varying radius of the constant camber deck.

Centerpoint Boundary Surface

Parents and characteristic properties

Center point = control point at center of surface
Bounding curves = point entity or curve entity. four bounding curves

Description

The Centerpoint Boundary Surface is generated by a two-way interpolation between opposite edge curves. The shape is determined primarily by the edge curves, which are blended together in an interesting and usually attractive and useful way. The data point (Center point), which the patch interpolates at parameters u = .5, v = .5, provides additional control over the patch interior.

Bounding curves can be any combination of four curve and / or snake entities. In addition, you can use a point, bead, magnet, or ring for one (or more) of the four curves. Use of a point entity for one edge forms a 3-sided patch that spans the three remaining edges.

The four curves nominally connect at their endpoints. The order of the four snakes is curve1, curve2, curve3, curve4; and curve4 connects back to curve1. The curves (including any point used for an edge) must be picked in sequence, going around the loop in either a clockwise or counterclockwise direction.

The Centerpoint Boundary Surface meets all four boundary curves, if these curves actually join end-to-end in sequence to form a closed loop. The four edges are identified as follows:

curve1 v = 0
curve2 u = 1
curve3 v = 1
curve4 u = 0

The u=0, v=0 corner of the surface is where curve1 and curve4 join. The u parameter is the same as the t parameter of curve1 (or 1 - t, if curve1 is used in reverse).

It is assumed that the boundary curves meet each other at the corners. However, this condition is neither enforced nor checked. If one or more corners are “open,” you still get a continuous surface patch, but it no longer meets all the edges.

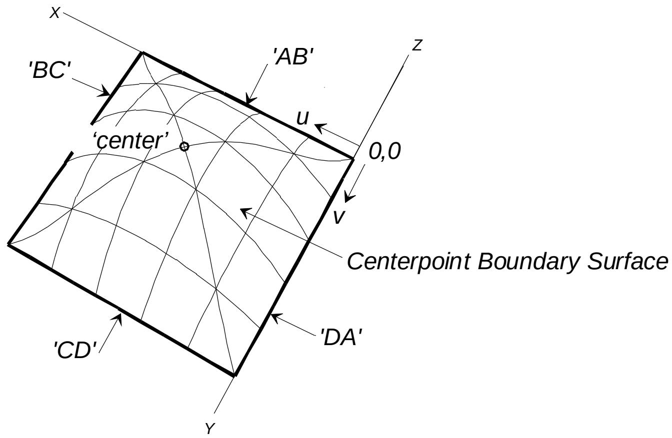
It is permissible, and often useful, to allow one edge of a Centerpoint Boundary Surface to be a single point. This forms a triangular patch that spans the three remaining edges.

Troubleshooting tip: If the surface doesn’t fill the boundary as expected, or turns out looking like a spider web, the most likely problem is that the bounding snakes are not in sequential order. To fix this, Edit Definition and either pick the snakes again or use the Selection Set pane and / or buttons to reorder the list.

Example: CenterptBoundary

‘blend1’ is a flat cornered blended surface formed by the four edge curves ‘AB’ (blue), ‘BC’ (green), ‘CD’ (red), and ‘DA’ (yellow) plus the center point ‘center’ (black).

‘AC’ and ‘BD’ (magenta) are two Line Snakes lying on ‘blend1’. They have been included to show the surface shape near the corners in more detail.



CenterptBoundary

See also: Tangent Boundary Surface

CGPoint

Characteristic data

Name -- as with any entity
Color -- as with any graphic entity
Visible -- true / false
Layer -- as with any entity
Lock -- as with any entity
Entity List – List of entities with weight, *WEIGHTS can be used
Weight -- Not editable – grayed out sum of all entities in Entity List
User data -- as with any entity

CGPoint is a point-class entity type that encapsulates calculation of the total mass and center of gravity of a set of graphic objects. By a special convention, this set can be all the weighted objects in the model.

There is a new system object named *WEIGHTS. It can serve as a list, and when doing so, it means the list of all objects in the model with non-zero unit weight.

The CGPoint location is the center of gravity of the graphic objects in the expanded list (or in the entire model, in case the list is *WEIGHTS). The mass (and unit mass) of the CGPoint is the total mass of the same set of objects. Symmetry images are taken into account transparently in making the calculation.

Unlike all other graphic entities, unit weight is not an editable property of a CGPoint. The edit field in the Properties Manager could be omitted, but I'd prefer to have it grayed out so you could read the weight total there.

Composite Surface

Parents and characteristic properties

Log-tolerance = distance tolerance to decide if the component surfaces connect
Number of Triangles = number of triangles in composite surface triangulation
Surfaces = surface entity. surface1, surface2 . . . surfaceN

Description

A composite surface is a way to combine more than one surface into a single entity in a way similar to curves combining into a PolyCurve. The Composite Surface supports a continuous triangle mesh for purposes of flattening and triangle mesh exports.

The natural orientation of a composite surface is defined as follows:

If the composite surface is open, its orientation is that of the **natural orientation** of the first parent. All other faces get oriented consistently with the first face.

If the composite surface is closed (no holes), all normals point inwards. Note that a sphere or torus (doughnut) counts as closed because the surface skin has no holes.

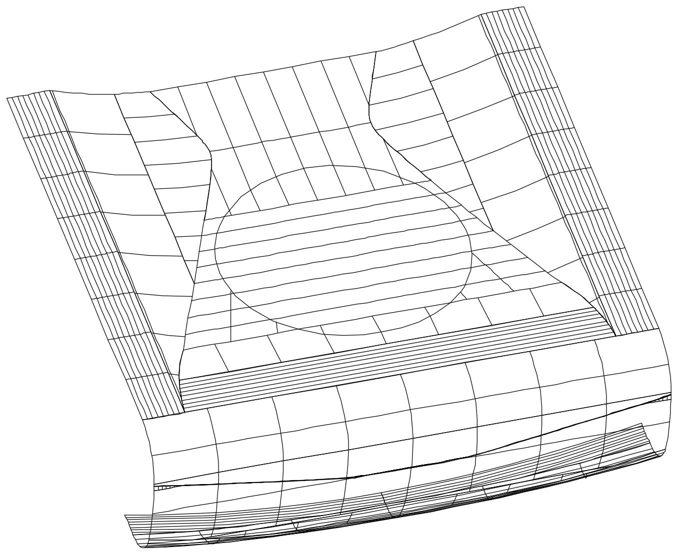
When setting up the list of surfaces be careful not to include a trimmed surface and its base surface since that would result in two superimposed surfaces.

Log-Tolerance is the log of the distance between surfaces that will be considered linkable. If Log-Tolerance is -3, for instance, surfaces that are within .001 units of each other will form a Composite Surface without error. If a surface is farther apart than the Log-Tolerance from its nearest neighbor, you will get error 424, "The parent surfaces are not all connected or are not all connected by edge adjacency."

Number of Triangles determines how many triangles the Composite Surface will contain.

Surfaces may NOT include either Triangular Meshes or other Composite Surfaces.

Example: TopCushionKnit2



TopCushionKnit2

TopCushionKnit2 is a representation of an airplane seat. It contains 7 composite surfaces each with 2 to 4 parent surfaces.

Conic Section

Parents and characteristic properties

center = point to locate center point of complete conic section (P)
point2 = point to locate end of primary axis (P2)
point3 = point to locate end of secondary axis (P3)
s0 = conic parameter (degrees) for the start (t = 0)
s1 = conic parameter (degrees) for the end (t = 1)
type = 1, 2, 3, or 4

type-1 = ellipse

type-2 = hyperbola, positive branch

type-3 = hyperbola, negative branch

type-4 = catenary

Description

All types of Conic Sections are constructed by specifying the center (center), the ends of the semimajor and semiminor axes (point2 and point3), and two parameter values (s0 and s1). The Conic Section always lies in the plane of the three points. The end of the primary axis is at point2 (P2). A temporary point Q is created for the end of the secondary axis by moving from P2 parallel to the primary axis as required to make the secondary axis perpendicular to the primary axis. Thus the user is freed from locating the three points such that P2-P-P3 is a right angle. The length of the secondary axis is the distance of P3 from the primary axis, also the distance from P to Q.

For a precise description of the Conic Section we define the three vectors using bold face letters:

a is the vector from P to P2 (primary axis)
b is the vector from P to Q (secondary axis)
c is the vector from the origin to P

a, b, and c are their magnitudes, respectively.

Ellipse (type-1)

For an ellipse, the primary axis can be either the semimajor or semiminor axis, depending on the relative distances of P2 and Q from P. The full ellipse passes through points P2 (at s = 0) and Q (at s = 90 deg.). The equation for the ellipse (type-1) is:

x(t) = a cos s + b sin s + c

s = s0 * (1 - t) + s1 * t

The ellipse foci are at **c** +/- **a** SQR(1 - b^2 / a^2).

Hyperbola (type-2 and type-3)

For a hyperbola, the primary axis is always the semimajor axis. The full hyperbola passes through point P2 (at s = 0).

The equation for the positive hyperbola (type-2) is:

x(t) = a cosh s + b sinh s + c

s = s0 * (1 - t) + s1 * t

The equation for the negative hyperbola branch (type-3) is:

x(t) = - a cosh s - b sinh s + c

s = s0 * (1 - t) + s1 * t

The hyperbola foci are at **c** +/- **a** SQR (1 + b^2 / a^2). The asymptotes are the two lines:

x = c + r * (a +/- b)

where r is a real parameter.

Catenary (type-4)

Although not actually a conic section, the catenary is a related analytic curve which is easily supported in the same context. Its equation is:

x(t) = a cosh s + b s + c

s = s0 * (1 - t) + s1 * t

(s in radians). If s0 and s1 are fairly small, the curve is very similar in shape to the positive hyperbola.

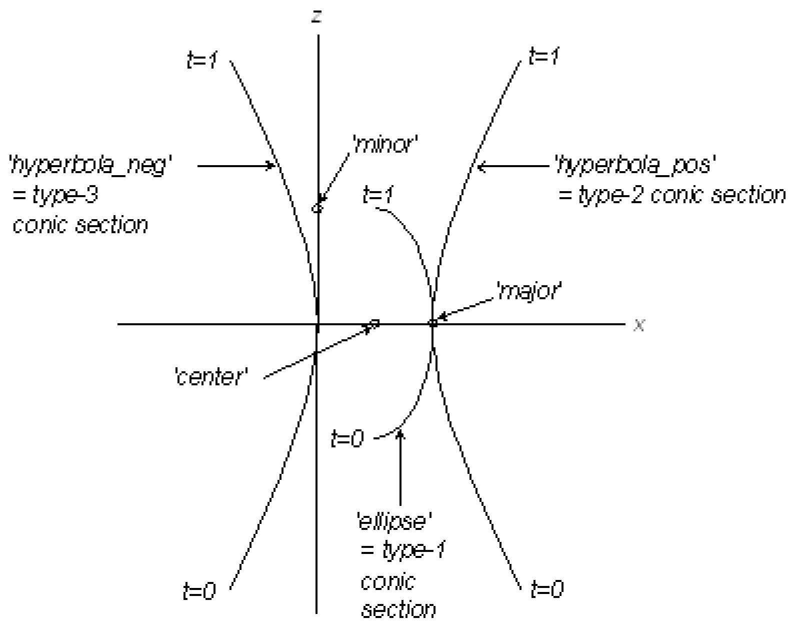
The catenary is the equilibrium curve assumed by a hanging chain or cable under its own weight, with the force of gravity acting in the **-a** direction. It has engineering applications in the design of bridges and arches.

For all types of Conic Sections, relabel is used to relabel the curve. The program's default relabel '*' produces the "natural" labeling, in which the parameter t is linearly related as indicated in the formulas above.

Note that the parabola is also a conic section. It can be regarded as the limiting case of either an ellipse or a hyperbola as the center moves to infinity. Parabolas can be created in SurfaceWorks in two ways — any type-2 B-spline or C-spline curve with three control points is a parabola.

Example1: Conics.ms2 (Ellipse and Hyperbola)

Example 1: conics.ms2 (ellipse and hyperbola)



CONIC.MS2 in <Y> view; Conic types 1 - 3.

All three of the conic sections in this example are formed using the same control points and conic parameters:

'center' is the center point

'major' is point2, the end of the primary axis

'minor' is point3, the end of the secondary axis

s0, the conic parameter for the start (t = 0) of the curve is -90

s1, the conic parameter for the end (t = 1) of the curve is 90

ellipse' is the type-1 conic section, 'hyperbola_pos' is the type-2 conic section, and 'hyperbola_neg' is the type-3 conic section.

Example 2 catenary.ms2 (catenary)

'catenary' is a type-4 conic section formed by the following three control points and two conic parameters:

‘center’ is the center point

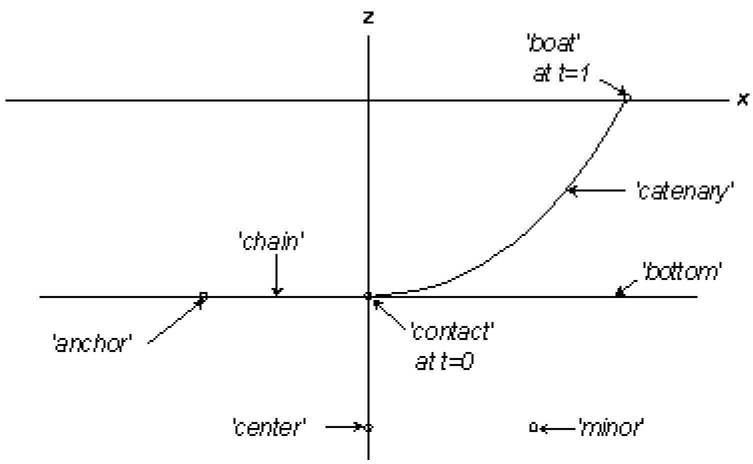
‘contact’ is point2, the end of the primary axis

‘minor’ is point3, the end of the secondary axis

s0, the conic parameter for the start (t = 0) of the curve is 0

s1, the conic parameter for the end (t = 1) of the curve is 90

In this example, ‘bottom’ is a line depicting the bottom of the bay (or lake or ...). ‘catenary’ is a type-4 conic; it is the portion of anchor chain not resting on the bottom. ‘chain’ is a Line; it is the portion of anchor chain that is resting on the bottom. ‘boat’ is a bead at the t=1 end of ‘catenary’.



CATENARY.MS2 in <Y> view.

See also: Arc, ArcSnake, BCurve, Ccurve

Contours

Parents and characteristic properties

Cut Type = Offset from Mirror / Surface
Mirror/Surface = a plane, surface, Line, or point entity
Index of first contour = index of first contour (an integer)
Index of last contour = index of last contour (an integer)
Signed distance from Mirror/Surface to 0-index contour = distance of the zero-index contour from the mirror
Contour spacing = contour spacing
Surfaces/Trimeshest = names of surfaces or trimeshes to be cut by contours

Cut Type = Normal to Curve
Curve = curve entity
Index of first contour = index of first contour (an integer)
Index of last contour = index of last contour (an integer)
Parameter value of 0-index contour = parameter value for index = 0 contour
Contour spacing (parameter space) = parameter interval between contour positions

Surfaces/Trimeshes = names of surfaces or trimeshes to be cut by contours

Description

Cut Type '**Offset from Mirror/Surface**' Contours creates one or more parallel sections in a user-specified arbitrary orientation. Each section cuts the surfaces named in the surfaces list, creating 3D polylines having the color and visibility designated.

Contours are specified by naming a Mirror/Surface that all contours are parallel to, the indices of the first and last contours (Index of first contour and index of last contour), the offset of the 0-index contour from the basis plane (Signed distance from Mirror/Surface to 0-index contour), the spacing between the contours (Contour spacing), and the names of surfaces the contours are to cut(Surfaces to cut).

Contours have both an offset from the Mirror (measured normal to the mirror, along the mirror's positive normal) and an index (its sequential number in the set of contours). Index of first contour and Index of last contour are index numbers you use to specify the set of contours to include in a contours entity.

Contour offsets from the Mirror are calculated from the equation

$$q = q0 + qint \times index$$

where:

index = Index of first contour, Index of first contour + 1, Index of first contour + 2, ..., Index of last contour (if only one contour in the set, Index of first contour = Index of last contour)
q0 = Signed distance from Mirror/Surface to 0-index contour
qint = Contour spacing
q0 and qint are positive in the direction along the positive normal to the Mirror

Mirror/Surface can be a plane, surface, Line, or point entity:

When Mirror/Surface is a plane, the contours are cut by planes. q is the distance of the contour cutting surface from the mirror plane.

When Mirror/Surface is a surface, the contours are cut by parallel offset surfaces. q is the offset of the contour cutting surface from the mirror surface.

When Mirror/Surface is a Line, the contours are cut by cylinders with the Line entity as axis. q is the radius of the contour cutting surface from the mirror Line.

When Mirror/Surface is a point entity, the contours are cut by spheres centered at the point. q is the radius of the contour cutting surface from the mirror point.

Cut Type '**Normal to Curve**' Contours creates one or more parallel sections normal to Curve. Each section cuts the surfaces named in the surfaces list, creating 3D polylines having the color and visibility designated.

Contours are specified by naming a Curve that all contours are normal to, the indices of the first and last contours (Index of first contour and index of last contour), the parameter value of the 0-index contour (Parameter value of 0-index contour), the spacing between the contours (Contour spacing) in units of t, and the names of surfaces the contours are to cut(Surfaces to cut).

Contours have both a t-position along Curve and an index (its sequential number in the set of contours). First index and last index are index numbers you use to specify the set of contours to include in a contour object.

t-positions for the contours are calculated from the equation

```
t = t0 + tint x index
```

where:

index = first index, first index + 1, first index + 2, ..., last index (if only one contour in the set, first index = last index)
t0 = the parameter value for the index=0 contour (any value)
tint = the parameter interval between contour positions (any value)

For details on indices and on specifying a set of contours, see "Understanding SurfaceWorks Entities - Collective Entity Information - Contours Entities”.

Note that contours cannot be used for constructing other entities. If you want a curve that can be used in defining another entity, use a Projected Snake or an Intersection Snake.

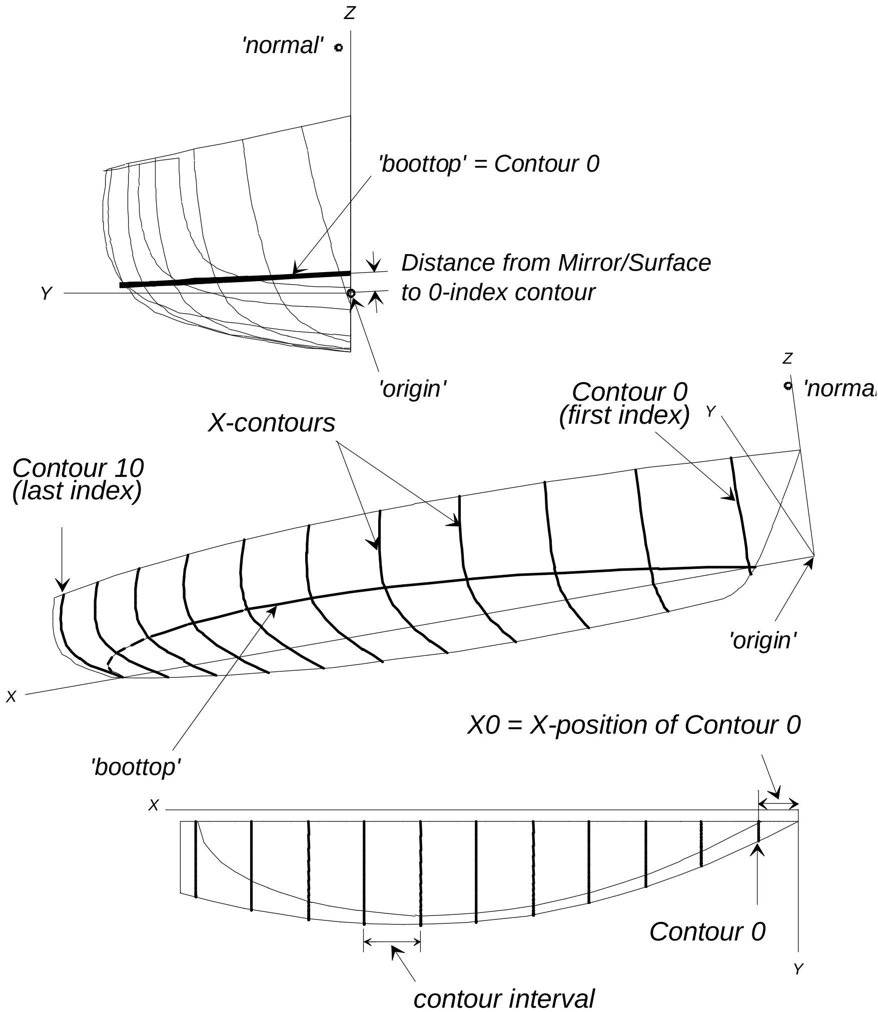
Example: contour1

‘hull’ is a C-spline Lofted Surface.

The Mirror / Surface for the Contour 'boottop' is ‘cutplane’, a 2-Point Plane (white) defined by the 2 points ‘origin’ and ‘normal’ (both white). The plane includes the X-axis and is inclined at about 3 degrees from horizontal.

‘boottop’ (white) is a single contour (Index of first contour = Index of last contour) parallel to ‘cutplane’ and .4 units above it (Signed distance from Mirror/Surface to 0-index contour = 0.4). We used the convenience of Index of first contour = 0, Index of last contour = 0, and Contour spacing = 0 in specifying this single contour.

‘stations’ (cyan) is a family of 11 contours (Mirror/Surface = *X=0 , Index of first contour= 0, Index of last contour = 10, Distance from Mirror/Surface to 0-index contour = 1.95 and Contour spacing = 2.73) with the contours at X = 1.95, 4.68, 7.41, ... 29.25.

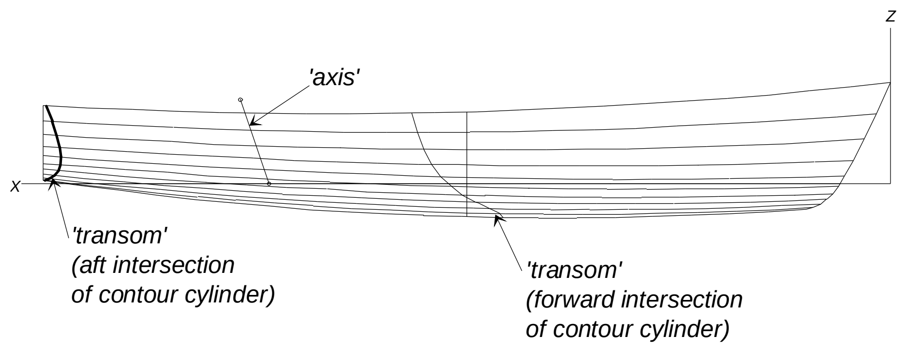


Contour1

Example: contourc

In this example, the Mirror/Surface for the contour entity is a line ('axis'; yellow) rather than a plane, and the contour is therefore cylindrical rather than a plane section.

'px' and 'py' are the endpoints for the mirror line 'axis', which lies in the centerplane and establishes the rake of the transom cylinder contour 'transom' (white) — it is inclined about 20 degrees from the vertical. Signed distance from Mirror/Surface to 0-index contour is used to specify the cylinder radius, in this case 7.5 ft. While Index of first contour = Index of last contour (both zero), make a single contour in this set, there are two intersections of the contour cylinder with the hull — the aft one is the transom we want, the forward one is extraneous. (If you want to get rid of the forward one, you'll need to erase it in your CAD program.)

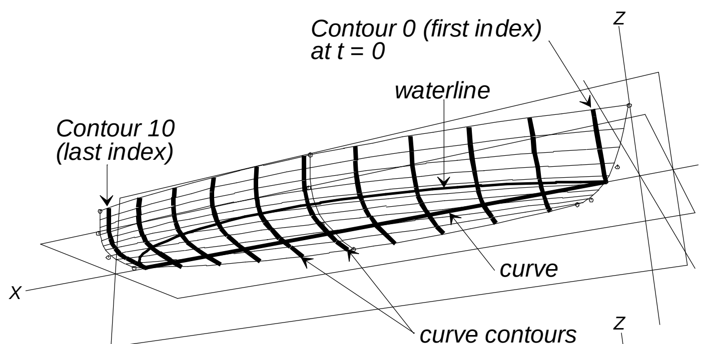


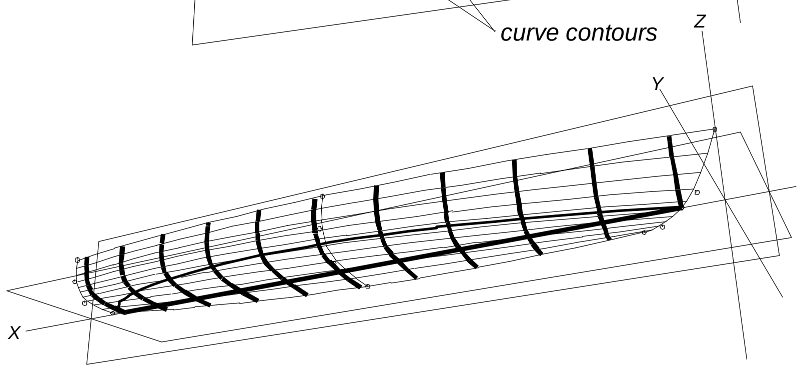
Contourc.

Example: CurveCont1

In this variant of the DEMO model, the stations are a set of Cut Type 'Normal to Curve' Contours (red) that durably divides the waterline into 10 intervals. The waterline (magenta) is an Intersection Snake (of hull and the Z=0 plane), with rings added at both ends. Each ring is projected onto the centerplane, and Curve (white) is the Line between the two Projected Points. The combination of 'Parameter value of 0-index contour' = 0 and 'Contour spacing' = .1 divides the line into 10 equal parts, and a contour is cut at each of these places. In this case, the cutting planes are all parallel because Curve is a straight line.

You can make lots of changes in the hull (even submerging part of the transom) and the stations adjust themselves, maintaining 10 equal intervals along Curve; for example, in the bottom figure, the hull has been stretched by X-scale = 1.5.





CurveCont1.ms2: initial model (top) and model with X-scale = 1.5 (bottom). In both cases, the t interval between contours is 0.1.

See also: Projected Snake, Intersection Snake

Contour Curve

Parents and characteristic properties

Type = 0 or 1
Contours = a contours entity
Index = a particular index
Mirror/surface = plane, surface, Line, or point entity. basis cutting plane or surface
Log tolerance = construction accuracy required

Description

The Contour Curve entity type provides a way to turn one cut of a contours-class entity into a parametric curve, which can then be used for further constructions. It has some of the characteristics of an Intersection Snake, since it can often be viewed as the intersection of a surface with an implicit surface such as a plane. However, unlike a snake, it has the advantage of being able to cross multiple surfaces.

Relabel redistributes the t parameter, as for other curves. (The default labeling for a Contour Curve is by arc length along the curve.)

Type (0 or 1) In conjunction with the Mirror / surface parent, determines forward (0) or reverse (1) orientation.

Index A particular index, in the index range of the parent Contours entity. This selects which of the Contours' cuts will be used to make the curve. (Note, this index is a real number, so it can be a Variable or the result of a Formula.)

Mirror/surface This parent is used solely to control the orientation of the curve, i.e., which end is t = 0.

Log-tolerance Accuracy requirement for the assembly of two or more cuts into a single curve.

Orientation procedure: when the separate cuts at this Index have been assembled end-to-end into a single curve, we are left (generally) with two free ends. (The exception is when the cuts assemble into a closed curve.) We calculate the signed distances of the two end points from the Mirror / surface parent.

For type 0, we orient the curve so t=0 is the end with the smaller signed distance.

For type 1, we orient the curve so t=0 is the end with the larger signed distance.

Note: It's always worthwhile giving some thought to the choice of the Mirror / surface parent. If the two ends of the assembled curve have nearly the same signed distance from Mirror / surface, then small changes in the model can flip the orientation of the Contour Curve, which might radically change downstream geometry.

Example

Contour Curve.MS2 - The Contour Curve is created from index number 4 of the contour set 'stations'.

Copies

See Copies in Reference Guide to Functions.

Copy Bead

See: Copy Ring

Copy Contours

Parents and characteristic properties

Type = 0 or 1
First Index =
Lasr Index = a particular index
Basic Contours = a contours entity
Source = surface
Destination = expanded surface or trimesh

Description

The Copy Contours entity type is a way to transfer contours from one surface, Composite Surface, or Trimesh to the corresponding locations on a different surface or Trimesh. It was principally motivated by the need to display X, Y or Z contour markings transferred from the 3-D basis surface, on Expanded Surfaces and Expanded TriMeshes.

It is not limited to these entity types, in case other useful applications of contour transfer should emerge.

Type determines how indices are obtained:

type-0 use the indices of the parent contours entity (ignore First index and Last index)

type-1 use First index and Last index from this entity

First Index must be in the index range of the Contours entity

Last Index must be in the index range of the Contours entity

Source must be one of the surfaces cut by the parent Contours entity

Destination nominally an expanded surface or trimesh made from the source

If the source is a surface (including Trimmed Surface, but not Composite Surface), any surface can serve as the destination. If the source is a TriMesh or Composite Surface, the destination must be a TriMesh that is congruent with the source, i.e., it has the same number of nodes and triangles, with the same indexing, order and connectivity.

Examples

CopyContours1.MS2 - Index numbers 2 through 5 of the station contours are copied to the pattern for a topside plate.

Copy Curve

See: Copy Surface

Copy Magnet

Parents and characteristic properties

Magnet = a basis magnet
Surface = the destination surface
u-graph = a graph
v-graph = a graph

Description

A Copy Magnet is a copy of a magnet or ring from the u,v-space of a “source” surface to the u,v-space of a “destination” surface. The u,v-spaces are the two “environments”.

In its simplest form, with both graphs equal to the default graph ‘*’, the Copy Magnet reads the u,v position of the basis magnet, and locates itself at that same u,v position on the destination surface. The graphs allow the u and/or v parameters to be transformed in the process, similar to the effects of the graph of a Copy Bead. Thus the Copy Magnet is “slaved” to the position of its basis magnet, with an optional transformation in between. The Copy Magnet has utility for making point type Procedural Surfaces, similar to the use of Copy Beads and Copy Rings for the other Procedural entities. That is, by use of a single Magnet and Copy Magnets slaved to it, you can easily move the construction to various u,v positions, verifying that the construction takes place as expected, or locating and “debugging” regions where the construction fails.

Example: Copy Magnet.MS2

Copy Point

See: Copy Surface

Copy Ring

The Copy Bead and Copy Ring are very similar to one another. As with other similarly named pairs of beads and rings, the only difference is that a Copy Ring, residing on a snake, also marks a position on the snake’s host surface, and so it can serve in any role where a magnet is needed. Typically, if the support is a snake, we would always make a ring rather than a bead.

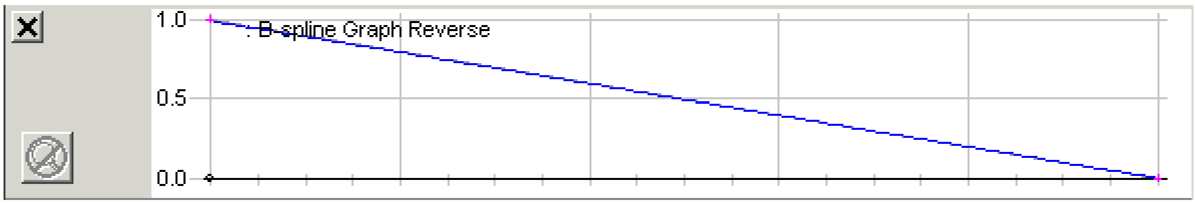
In the case of beads and rings, the two “environments” are the t parameter spaces of two curves or snakes. The data for a Copy Bead is a basis bead (or ring), a destination curve, and a graph.

In its simplest manifestation (with graph = ‘*’), a Copy Bead is a bead on a curve, ‘curve1’, which copies the t position of the basis bead (or ring) on its host curve (or snake) ‘curve0’. We think of the Copy Bead as being “slaved” to its supporting bead, often functioning as a “remote control”.

For the simplest example, picture two Lines ‘line0’, ‘line1’ (both with default relabel); an Bead ‘e0’ on ‘line0’, and a Copy Bead ‘e1’ made by copying ‘e0’ onto ‘line1’. As you drag ‘e0’, ‘e1’ will continuously update to remain at the same t location, and therefore the same proportional position along its line, as ‘e0’.

The graph support greatly enriches the behavior of Copy Bead and Copy Ring by permitting a transformation of the parameter between the source curve and destination curve. The default graph ‘*’ in this context signifies no transformation, i.e., h(t) = t. A more complex graph produces a more complex motion or response of the Copy Bead. Since there are few limits on the complexity of a graph you can specify practically any relationship between the positions of the two beads.

A simple graph example that is frequently useful is the type-1 BGraph with values {1, 0}, i.e., h(t) = 1 - t . We usually give this graph the name ‘reverse’. If introduced into the above example of two lines, it makes the Copy Bead go to the complementary position 1 - t, i.e. it traverses ‘line1’ from 1 to 0 as the basis bead ‘e0’ goes from 0 to 1.



Copy Beads and Copy Rings are frequently useful in the related topics of animation and procedural entities. A Procedural Curve, Procedural Snake or Procedural Surface that uses 2 or more driving beads is difficult to investigate because you have to move all those beads to new positions to visualize the construction taking place at any particular location. If you replace all but one driving bead with Copy Beads, you can easily slide the whole construction along the full length of the driving bead’s supporting curve and see what’s going on. If the construction fails at some position as you drag, the moving point or curve will disappear; if you accept the drag at that position, you’ll be able to see exactly how the construction fails, by the set of errors that pop up in the Error Dialog.

Animation also benefits from Copy Beads and Copy Rings. The sample file PITCHPOLE.MS2 illustrates this. Open the file, and drag Bead “go” around the circular arc. (Pitchpoling occurs when a vessel is caught by a steep breaking sea that somersaults her, head over heels. Like a collision at sea, a pitchpole can ruin your whole day.)

This model uses about a dozen Copy Beads, one Copy Ring, and several Copy Curves and Copy Surfaces. If you turn on layer 1 you can see how it works. The Copy Beads (bright red) are all slaved to the Bead ‘go’. They are control points for a BCurve ‘constr_wave’ (bright green), which creates the evolving wave profile. If you drag ‘go’ around the arc, you will see each Copy Bead slide correspondingly down its supporting curve. The supporting curves are mostly uniformly labeled lines; a few near the wave crest are curved or have relabels to control the details of the wave profile.

A Copy Curve ‘waveside’ is used to make a 1:1 copy of ‘constr_wave’ down at sea level; the sea surface is Translation Surface ‘wave’ made from ‘waveside’ and ‘wavesweep’. The motion of the boat, in relation to the wave crest, is controlled by Copy Ring ‘r0’ (also slaved to ‘go’), that slides along a Line Snake ‘n0’ on ‘wave’. Frame3 ‘F0’ is located at ‘r0’ and arranged to have its x and y axes tangent to the 'wave' surface at all times. The boat surfaces are several Copy Surfaces made from a base boat model positioned at the origin (on layer 2). Note that the boat is white in color; if you are not seeing it, make sure that your background is not white as well.

Examples

COPYBEAD1.MS2, COPYBEAD2.MS2, COPYRING1.MS2

Copy Snake

Parents and characteristic properties

Snake = parent (original) snake
Magnet/Surface = magnet, ring or surface. specifies parent surface and reference point for Copy Snake
Scale = scale factor in u,v-space
Angle = rotation angle in u,v-space (degrees)

Description

The Copy Snake is a copy of a parent snake onto the same or a different surface. The copy can be shifted, scaled, and rotated during the copy process by using Magnet, Scale, and Angle. For the plain (not shifted, etc.) copy, each point of the Copy Snake has the same u,v position on its host surface as has the corresponding point of the parent snake on its host surface. (If such a Copy Snake were on the same surface as the parent snake, it would be exactly the same as its parent snake.) For a shifted, scaled, and / or rotated copy, the reference point is specified by Magnet/Surface.

Magnet/Surface can be any kind of magnet or surface entity:

When Magnet/Surface is a surface, the Copy Snake will lie on that surface, and the surface's center in u,v-space (u = .5, v = .5) will be the reference point for the Copy Snake.

When Magnet/Surface is a magnet, it specifies the parent surface the Copy Snake is to lie on (the surface that magnet lies on), and it serves as the reference point for the shifted Copy Snake. With Scale = 1 and Angle = 0, this means that each point of the Copy Snake would be offset from its surface's center (u = .5, v = .5) by the u,v amount that the reference magnet is offset from the surface's center.

Scale is a scale factor for the copy, relative to the reference point. The default is 1.

Note: The scale factor is applied in u,v-space, and the actual size of the Copy Snake relative to the parent snake in X,Y,Z space will depend on the relative sizes of their parent surfaces.

Angle is a rotation angle for the copy, 0 – 360 degrees, relative to the reference point. The default is 0. The rotation is applied in the u,v-space of the parent surface. Positive rotation is counterclockwise when you are looking at the u,v-plane with u-axis to the right and v-axis up.

Thus, if these three conditions pertain:

Magnet/Surface is a surface, or a magnet at u = 0.5, v = 0.5
Scale is 1.000

Angle is 0 (or a multiple of 360 degrees)

then the Copy Snake and its parent snake are identically positioned in the u,v-spaces of their respective surfaces (although the actual sizes of the snakes will only be the same if their parent surfaces are of equal size).

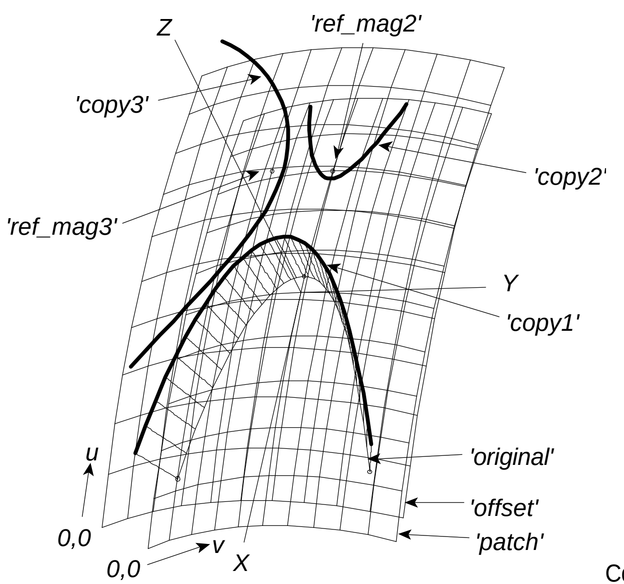
Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t labeling corresponds point by point to that of the basis snake.

Example 1: CopySnake1

'original' (magenta) is a snake on 'patch' (cyan). Three Copy Snakes, 'copy1', 'copy2', and 'copy3', are made from it.

'copy1' (red) lies on the Offset Surface 'offset' (green). It is positioned and scaled exactly like the parent snake 'original'. The correspondence of 'original' and 'copy1' is demonstrated by B-spline Lofted Surface 'ruled' (yellow) made between them — all of its u-constant mesh lines lie along normals to 'patch'. (Since 'offset' is a larger surface than 'patch', 'copy1' physically is larger than 'original'.)

'copy2' (blue) lies on 'patch' (the same surface as 'original'). It is positioned by Magnet 'ref_mag2', scaled by a factor of 0.5, and rotated 180 degrees.

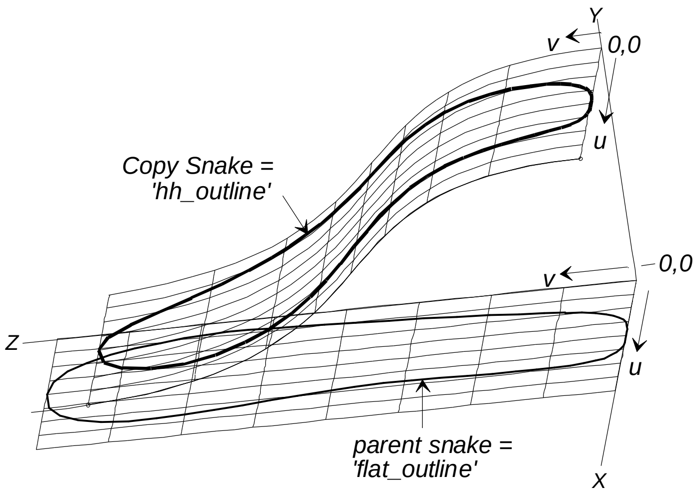


CopySnake1

'copy3' (black) lies on 'offset'. It is positioned by Magnet 'ref_mag3' and rotated 270 degrees.

Example 2: CopySnake2-ArcLengthRing

This model copies a snake from a flat surface onto a surface that has the same width and arc length as the flat surface (7 inches). 'flat_outline' is the parent B-spline Snake. 'hh_outline' is the Copy Snake; it uses direct specification of its surface, Scale = 1, and Angle = 0. To illustrate the difference between a Copy Snake and a Projected Snake, we've included a Projected Snake 'comparison' (blue).



CopySnake2-ArcLengthRing (Projected Snake not shown).

See also: Projected Snake

Copy Solid

See: Copy Surface

Copy Surface

Copy Point, Copy Curve, Copy Surface, Copy Solid

Point or Curve or Surface or Solid = basis entity
Frame1 = source frame
Frame2 = destination frame
x-scale, y-scale, z-scale = scale factors for the 3 frame axes

Each of these entities takes a support of the same class (the basis object); two frames; and three scale factors. The two “environments” are two frames, designated Frame1 (the “source” frame) and Frame2 (the “destination” frame). The process of constructing the Copy object has three steps:

1 transform the basis object into frame coordinates in Frame1

2 apply the three stretching factors to the x, y, z coordinates in Frame1

3 create the Copy at those same x, y, z coordinates in Frame2

All this sounds like a lot of Frames. Although Frames are versatile and not hard to create, remember that any point can serve as a frame; its x, y, z axes are parallel to the global X, Y, Z axes respectively. In practice, some 90% of Copy objects will use points for one or both frame supports.

The simplest application of Copy entities is to create a 1:1 copy of an object, parallel to its original orientation, but at another location. This is the best way yet to make a parallel copy of a curve. Suppose:

1 the basis curve is ‘c0’,

2 there is a point ‘pt0’ at its t = 0 end, and

3 there is a point ‘pt1’ where the copy is supposed to start.

Make a Copy Curve from ‘c0’ with Frame1 = ‘pt0’ and Frame2 = ‘pt1’, and 1, 1, 1 scale factors. The two frames are parallel in this case, so the transformation is just a parallel displacement equal to the displacement from ‘pt0’ to ‘pt1’.

If the Copy needs to be the same shape, but a different size, you can apply equal scale factors, e.g. 0.5, 0.5, 0.5 for a half-scale curve or surface. If the copy needs to be scaled differently in 2 or 3 directions, use unequal scale factors. To make a mirror image, one or more of the scale factors can be negative -- or the destination frame needs to have the opposite orientation (“handedness”) from the source frame.

If the Copy needs to be in an entirely different location and orientation, then build a frame in the required position, and use it as the Frame2 support.

There’s no requirement that Frame2 be different from Frame1. Sometimes a Copy into the same frame is useful. Suppose you want a duplicate of a surface in the same place, with no change except different color, different divisions, or a different layer? Then you can use the same point (any point, in fact) for Frame1 and Frame2.

Copy Point provides the best way yet to make the fourth corner of a rectangle (or parallelogram) when three corners are known. Suppose the three known corners are ‘pt1’, ‘pt2’, ‘pt3’. Then the corner opposite to ‘pt2’ is the Copy Point made with ‘pt1’ as basis point, ‘pt2’ as source frame, and ‘pt3’ as destination frame. (This is more convenient than a Blend Point, because you don’t have to mess around with any weights.)

Examples

Copycurve.MS2, RPY.MS2, RPY-JET.MS2, CopySolid1.ms2

Copy TriMesh

Parents and characteristic properties

degree = degree of subdivision
Triangle mesh = parent TriMesh
Frame1 = parent location
Frame2 = copy location
Scale Factors, X, Y, and Z

Description

Copy entities takes a parent of the same class (the parent entity); two frames; and three scale factors. The two “environments” are two frames, designated Frame0 (the “source” frame) and Frame1 (the “destination” frame). The process of constructing the Copy object has three steps:

1 transform the basis object into frame coordinates in Frame0;
2 apply the three stretching factors to the x, y, z coordinates in Frame0;
3 create the Copy at those same x, y, z coordinates in Frame1.

All this sounds like a lot of Frames. Although Frames are versatile and not hard to create, remember that any point can serve as a frame; its x, y, z axes are parallel to the global X, Y, Z axes respectively. In practice, some 90% of Copy entities will use points for one or both frame parents.

Example CopyTriMesh1.ms2

'tm' (the parent TriMesh) is a 5-node, 4-triangle TriMesh entity similar to TriMesh1.ms2. 'I0' is a Copy TriMesh made from 'tm', with the two points 'p3' and 'p6' as the two frames. 'I0' has an x- scale factor of 0.5, so it is only half as wide in the X-direction as its parent.

See also: Copy Curve, Copy Surface

C-spline Curve

Parents and characteristic properties

Degree = C-spline degree: 1 = linear, 2 = quadratic, 3 = cubic
Control points = point entity. point1 . . . pointN

Description

A C-spline Curve is a continuous curve defined by a series of control points. The curve ends at the two end control points (point1 and pointN) and passes through the others in sequence.

degree-1: the curve is the polyline itself (Fig. 1, top)

degree-2 and up: the curve behaves more and more "stiffly" (Fig. 1, middle and bottom)

For details about C-spline degrees and how they behave, see "Understanding SurfaceWorks Entities - Collective Entity Information - C-spline Entities".

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is approximately uniformly distributed with respect to chord length along the polyline.

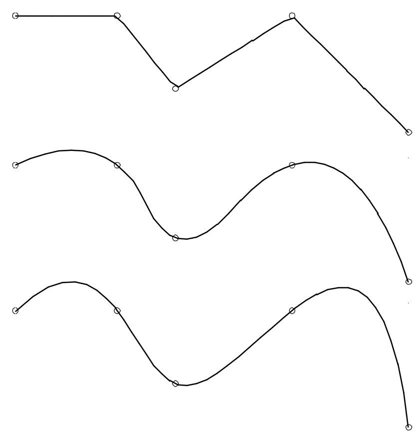


Fig. 1 Three degrees of C-spline Curve

Example: CCurve

'curve' is a parametric cubic spline (degree-3) passing through the six parent control points: 'xpt1', 'xpt2', 'xpt3', 'xpt4', 'xpt5', 'xpt6' in sequence (Fig. 2).

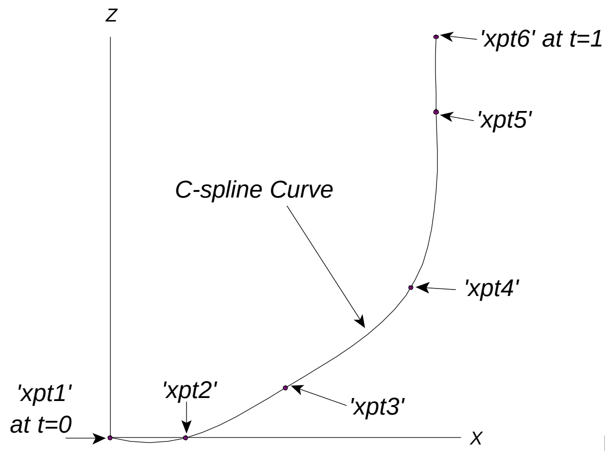


Fig. 2. CCurve.

See also

C-spline Snake
C-spline Lofted Surface
B-spline Curve

See also: C-spline Snake, C-spline Lofted Surface, B-spline Curve

C-spline Lofted Surface

Parents and characteristic properties

Relabel = Relabels surface in lofting direction
Degree = longitudinal C-spline degree, 1 to 3
Control curves = point entity or curve entity. curve1 ... curveN

Description

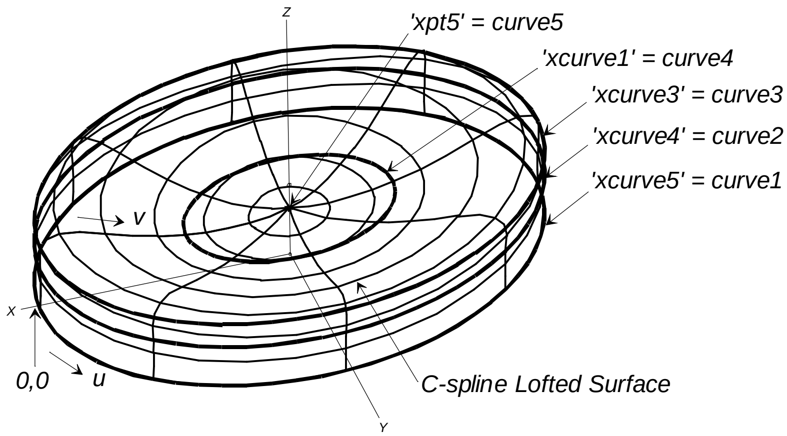
In the C-spline Lofted Surface, the lofting curves are C-splines. This means the C-spline Lofted Surface passes through all of its control curves. For details about how lofted surfaces are generated, see "Understanding SurfaceWorks Entities - Collective Entity Information - Lofted Surfaces". For details about the three C-spline degrees and how they behave, see "Understanding SurfaceWorks Entities - Collective Entity Information - C-spline Entities".

The u parameter on the surface is the same as the t parameter for the first control curve. The v parameter is the parameter along the lofting C-splines, approximately proportional to arc length. The lofting C-splines are the u = constant lines.

Relabel is used to control the labeling of the v-direction (the lofting curves) of the surface.

Example: CLoftSurf

This C-spline Lofted Surface (blue) is made from 5 imported parents: 'xcurve5', 'xcurve4', 'xcurve3', 'xcurve1' and point 'xpt5', selected in that order. The C-spline Lofted Surface passes through all of its control curves. For comparison, see the B-spline Lofted Surface example which uses the same set of control curves as parents, but the surface only passes through the first and last control curves. Also notice that while in the B-spline Lofted example the model is tangent across the center, in the C-spline Lofted model, there is a little peak in the center.



CLoftSurf

See also: Arc Lofted Surface, B-spline Lofted Surface, C-spline Curve, Demo.ms2

C-spline Snake

Parents and characteristic properties

Degree = degree of C-spline: 1 = linear, 2 = quadratic, 3 = cubic
Control magnets = magnet or ring. magnet1 . . . magnetN

Description

A C-spline Snake is a continuous curve lying in a surface and defined by a series of magnets for control points. The magnets must all lie in the same surface. The curve ends at the two end magnets (magnet1 and magnetN) and passes through the others in sequence. In the u-v parameter space, the C-spline Snake is a true C-spline Curve.

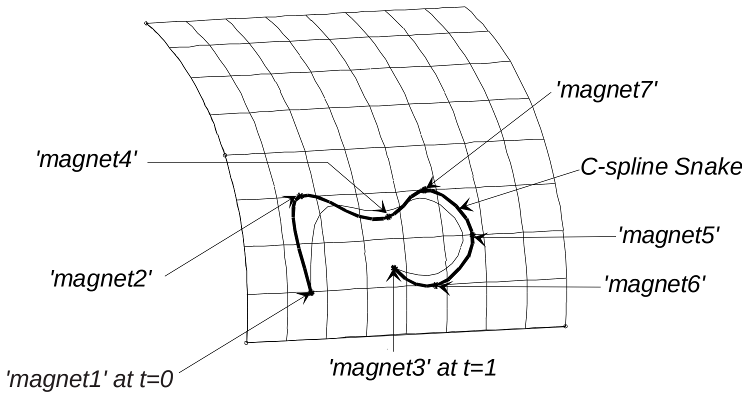
For details about the three C-spline degrees and how they behave, see “Understanding SurfaceWorks Entities - Collective Entity Information - C-spline Entities”.

Relabel is used to relabel the curve. The program's default Relabel ‘*’ produces the "natural" labeling, in which the parameter t is distributed approximately uniformly with respect to chord length along the C-spline Snake, in the u-v parameter space.

Example: CSnake

The degree-2 C-spline Snake (dark cyan) uses magnets ‘magnet1’ through ‘magnet7’ as its control points. The c-spline curve passes through all its control points. The other curve (magenta), for comparison, is a degree-2 B-spline Snake using the same set of control points; it passes through only its first and last control points.

‘magnet1’ is a Magnet (with absolute u,v coordinates). All the other magnets are relative to 'magnet1' or to each other. Magnets 2, 3, 5, and 6 are based on ‘magnet1’. Magnets 4, and 7 are based on magnets 3, and 6 respectively and are positioned to roughly mirror their parent points across the centerline of the keyhole. The set of control points and the snakes can be moved anywhere on the surface simply by moving ‘magnet1’. If the surface is modified in any way, the snakes will change their position in space, but will always remain on the surface in a corresponding position.



CSnake

See also: B-spline Snake, C-spline Curve

Developable Surface

Parents and characteristic properties

Control beads / curves = bead, curve, ring or snake. edges (bead or ring = edge designator)

Description

A Developable Surface is a special ruled surface which can be rolled out flat onto a plane without stretching of any portion. Developable Surfaces are highly desirable for fabrication of metal boats, as bending of a flat sheet is such a simple operation compared with the in-plane stretching necessary to form compound curvature.

Given two curves, curve1 and curve2, Developable Surface finds a set of rulings connecting the two curves and satisfying the conditions of developability. The procedure is similar to the graphical process developed by Kilgore. If the curves are complex, dissimilar in shape, or have inflection points, the Developable Surface may consist partly or entirely of cones having their apices on one curve or the other.

Each curve can be specified by either a bead or a curve name:

When bead/curve is a bead, it serves two purposes:

1. It tells which curve is selected for this edge of the Developable Surface (the curve that bead belongs to).
2. It provides a starting point for the search for the end of one ruling.

When bead/curve is a curve, the effect is the same as a bead located at t=0.5.

In some cases, there is more than one possible developable surface spanning two given curves. In this situation, by using one or two beads to suggest where the first ruling should go you can influence the startup of the ruling process and thereby select which of the multiple solutions the program will find. Note that you usually cannot arbitrarily prescribe any single ruling; the program will just take your suggested ruling as starting points for its search, and will search from there for the nearest ruling that joins the two curves and satisfies the developability conditions. Then it fills in the remaining rulings by stepping toward each end of the panel.

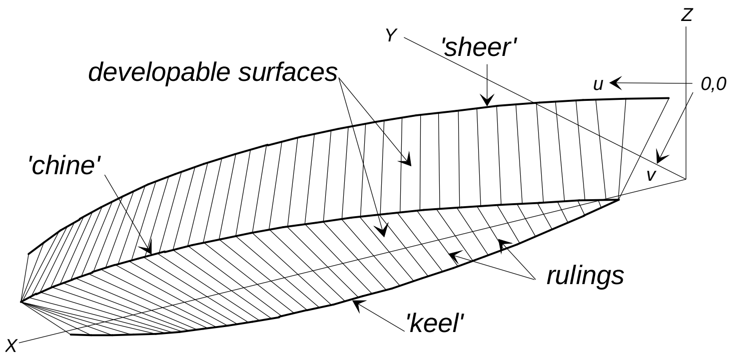
It is important that the two curves be correctly oriented. The u = 0 edge of the surface will be a ruling connecting the start of curve1 with the start of curve2, and the u = 1 edge will connect the ends of the two curves. If the two curves have opposite orientation, you can use SubCurve or SubSnake to reverse the orientation of one.

v is a linear parameter along the rulings, running from v = 0 at curve1 to v = 1 at curve2.

If you will be using MSDEV to expand Developable Surface panel shapes, be sure that the normal orientation is set correctly. MSDEV uses the orientation to determine the direction of thickness for panels — thickness runs in the same direction as the normal orientation. For information on figuring out the direction of a surface's normal, see "Basic Concepts - Parametric Surfaces"

Example 1: devsurf1

'panel1', the topsides panel on a single-chine sailing hull, is a developable surface specified by the 2 curves 'sheer' and 'chine'. 'panel2', the bottom panel of the hull, is specified by the 2 curves 'chine' and 'keel'. All three of the curves are type-2 C-splines. 'chine', as you may have noticed, is doing "double duty". The rulings for 'panel1' begin on 'sheer' and end on 'chine'; those for 'panel2' begin at 'chine' and end on 'keel'.

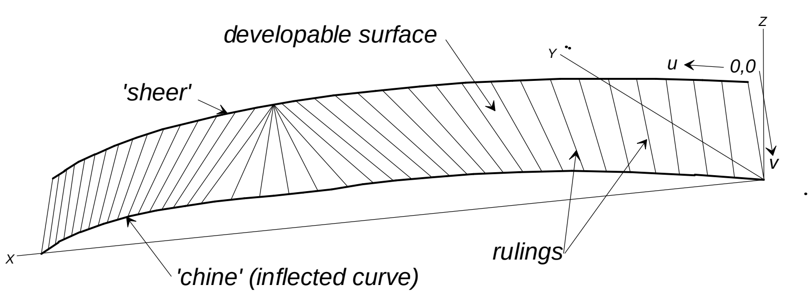


devsurf1.

Both panels are displayed showing only the u-constant lines (the rulings).

Example 2: devsurf2

In this case, 'panel1' is also specified by 'sheer' and 'chine', but here 'chine' is a type-3 B-spline curve with 2 inflection points.



devsurf2.

See also: Ruled Surface

Edge Snake

Parents and characteristic properties

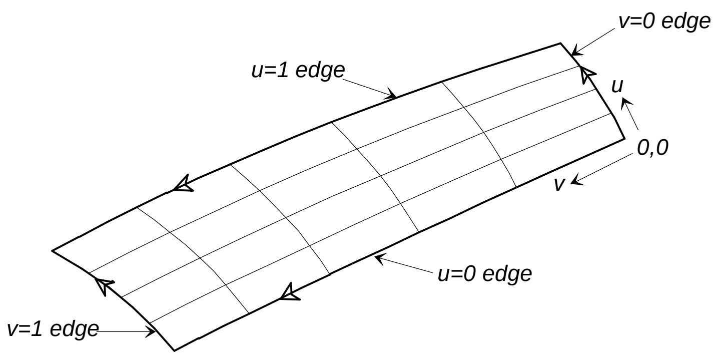
Surface edge = edge of surface the snake lies along
Surface = parent surface

Description

The Edge Snake is a special line snake lying along one complete edge of Surface and oriented either in the same or reverse direction as the surface parameter.

Surface edge specifies which edge of surface the snake is to lie along: v = 0, u = 1, v = 1, u = 0.

Fig. 1 shows the four types of Edge Snake in which t runs in the same direction as the surface's u or v parameter. The arrows along the Edge Snakes show the direction of increasing t. For the four reversed types, t would go in the direction **opposite** to the arrows.



The four types of Edge Snake in which t runs in the same direction as the surface's u or v parameter. The arrows along the Edge Snakes show the direction of increasing t. For the four reversed types, t would go in the direction opposite to the arrows.

A snake following an edge could also be formed by a Line Snake that joins two magnets at adjacent corners of the surface, or alternatively, by a UVSnake using a magnet located on one edge of the surface. When a snake has to lie along a complete edge, however, there are several advantages in using an Edge Snake:

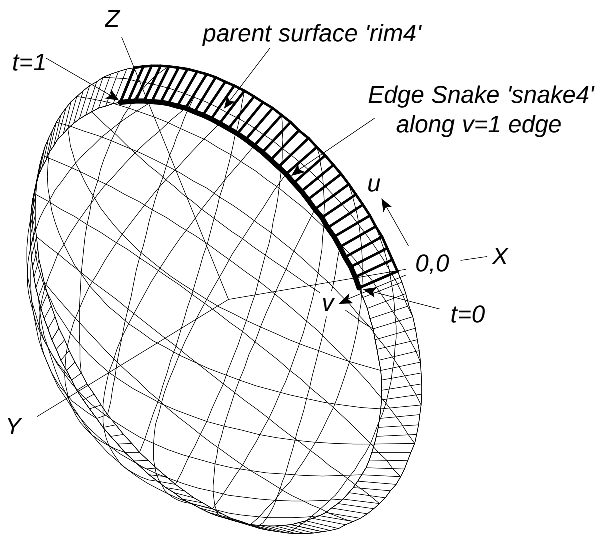
- it requires making only a single entity, rather than two or three
- the magnets you would place at the corners would often obscure other points already at those corners
- the properties of lying along the edge and traversing the entire edge are durable, not subject to location of magnets

Relabel The "natural" labeling of an Edge Snake has t equal to u for v=0 and v=1 snakes and t equal to v for u=0 and u=1 snakes. In other words, when v is constant, t is u; when u is constant, t is v. When the orientation is reversed t is 1-u for the v snakes and t is 1-v for the u snakes.

Example: EdgeSnake

In this example there are four Edge Snakes ‘snake1’ - ‘snake4’ (magenta), each located at the v=1 edge of one of the four B-spline Lofted Surfaces ‘rim1’, ‘rim2’, ‘rim3’, ‘rim4’ which span the inside circumference of the headlight rim.

The reflector of the headlight is a Tangent Boundary Surface ‘reflector’ that uses Edge Snakes ‘snake1’ - ‘snake4’ as its bounding curves. Tangency is maintained because edge tangency was specified for all four edges.



EdgeSnake

See also: Line Snake, UVSnake

Entity List

Parents and characteristic properties

Entities = list of parent entities (entities “contained” in the Entity List)

Description

The Entity List serves as a container for a list of entities. A sample usage of this entity is the saving and restoring of a named Selection Set. Creation of an Entity List is initiated by choosing **Insert>Entity List**. You can preselect the Entity List’s parents (that is, the entities contained in the Entity List), or you can select them via the Insert dialog. Any SurfaceWorks entity can be an Entity List parent, including other Entity Lists.

Actions performed on an Entity List are NOT iteratively performed on the entities which belong to (are parents of) the Entity List; e.g. Delete removes the Entity List entity only, not the entities in the list.

Although Entity Lists are not visible entities, you can select them via the Entities Manager or the Available Entities Manager.

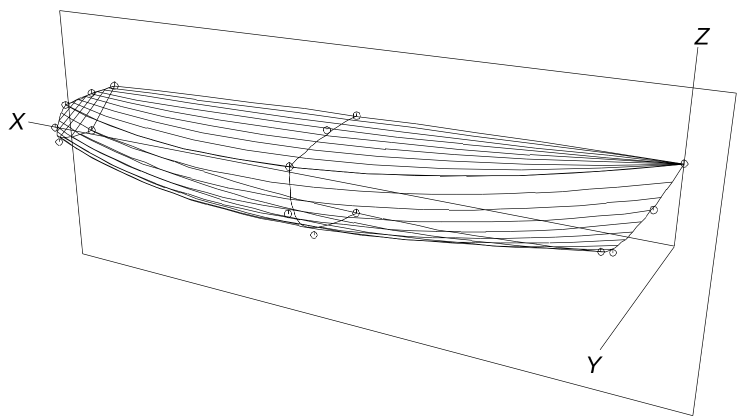
To put the Entity List parents into the Selection Set so you can perform some action on them (e.g. Edit>Show>Selection Set), you can choose Select>Parents>1st Generation. If you don’t want the Entity List itself to remain in the Selection Set, just click on it in the Selection Set pane (the Entity List will be the first entity in the Selection Set) and press ✕.

Example: entlist.ms2

Entlist is a sailboat model in which we have created some Entity Lists as aids for working on various portions of the model. By showing only the entities in an Entity List, we can refine that part of the model without the other parts of the model cluttering up the picture.

The Entity Lists in this model are:

‘HullDeckTransom’ includes the hull, deck, and transom surfaces and their parents (shown in the figure).
‘Keel’ contains the two keel surfaces and their parents.
‘BowMC’ contains the bow master curve and its control points.



entlist with only the entities in the Entity List ‘HullDeckTransom’ showing.

To show just the parents of an Entity Lists we would do the following:

Select the Entity List (from the Surface Manager)
Show-Hide>Hide Unselected
Show-Hide>Show Parents

Expanded Curve

Parents and characteristic properties

Orientation = pick axis onto which to expand the curve
Bead/curve = basis curve
Frame = frame for expansion
Scale = scale size if desired

Description

Orientation specifies which axis of the frame to use, and which direction along that axis (+/-) corresponds to the positive t direction.
Bead/curve specifies the basis curve; if a bead, it also specifies a particular point on the curve to end up at the frame origin.
Frame specifies the location of the frame
Scale scale factor for the expansion

An Expanded Curve is to a curve as a development (flattening) is to a developable surface: It is a copy of the curve, rolled out flat to make a straight line, with preservation of arc-length of all elements.

Note that this construction implies that, in absence of a relabel, the velocity profile of an Expanded Curve will be identical to that of its basis curve.

One application is the layout of structural elements that will be formed by bending straight tubes, shapes or extrusions into curves. In this case, the 3-D basis curve should follow the neutral axis of the shape.

Examples: ExpdCurve1.ms2

This is the forward master curve of Demo.MS2, flattened into a vertical line. Point 'p0' is used as the frame. The type is -3, therefore the positive-t direction on the ExpdCurve is the negative z axis, i.e., downward. A bead 'e0' at t = 0.257 specifies which point of the ExpdCurve should be at the origin.

Expanded Surface (Flattener authorization required)

Parents and characteristic properties

Type = expansion plane
Magnet/Surface = the basis surface
Magnet = used to orient the expansion in the plane of flattening
Thickness = thicknes of the material for the surface
Frame = location of expanded surface
Constraint = -1 or 0 or +1
Scale1 = scale factor for primary axis stretch
Scale2 = scale factor for secondary axis stretch

Description

The Expanded Surface is a surface-class entity that encapsulates conformal flattening of a parametric surface. This entity allows flat pattern development to take place entirely within the environment of SurfaceWorks, rather than saving a file and performing flattening with a separate program.

The Magnet/Surface specifies the surface to expand. It is expanded as follows: if a surface is specified for Magnet/Surface, the point u = 0.5, v = 0.5 is taken , if a magnet is specified for Magnet/Surface, then the magnet is used. This is placed at the origin of Frame. The surface is then expanded so that the magnet that is specified in Magnet lies along the Frame's axis as specified by Kind.

Kind

x-y plane(default) the Expanded Surface lies in the x,y plane of Frame, with the image of Magnet along Frame's x-axis

y-z plane the Expanded Surface lies in the y,z plane of Frame, with the image of Magnet along Frame's y-axis

z-x plane the Expanded Surface lies in the z,x plane of Frame, with the image of Magnet along Frame's z-axis.

Magnet/Surface the basis surface (an offset from it, actually, at half the thickness) is the surface to be flattened. The basis surface, and a point on it, are specified by the Magnet/Surface parent:

If Magnet/Surface is a magnet, the basis surface is the host surface of the magnet
If Magnet/Surface is a surface, then u=0.5, v=0.5 is used as the magnet location.

The image of this point is located at the origin of Frame.

Magnet is used to orient the flat pattern in the plane of expansion. The flattened surface is rotated so the image of this magnet is on the positive primary axis (as specified by Kind).

Constraint is an option, similar to MSPlex and Flattener, to apply an overall strain adjustment to the flattened surface, to compensate for limitations of the forming process.

constraint = -1: strain is constrained to be negative (contraction), in going from flat to curved
constraint = 0: no strain constraint. All edge lengths are accurately preserved
constraint = +1: strain is constrained to be positive (expansion), in going from flat to curved.

Scale1 and Scale2 are applied at the end. They just multiply the coordinates in the primary and secondary axis directions, respectively. One motivation for scale factors is fabric stretch factors in apparel and signage applications. Another is an observation that some users like to have a little safety margin; it's relatively easy to trim a plate that comes out a little large after forming, but it's a much bigger problem when one comes out slightly small.

A triangle mesh (roughly 2000 triangles) is made from the surface, and flattened by the trimesh flattening algorithm. Following flattening, the surface is approximated by a non-uniform B-spline surface. This is to provide the derivatives needed for Hermite interpolation, and for evaluation outside the [0,1]x[0,1] parameter square.

The strain distribution is calculated and saved internally, but is not currently visible to the user except through Strain Contours, Strain command, and STRAIN function.

(If the basis surface is already flat, the strain solution is bypassed; strain is identically zero, and constraint and thickness have no effect.)

Thickness is applied, similar to our other flattening methods. The actual surface triangulated for flattening is a uniform normal offset of the basis surface, in the direction of the basis surface positive normal when thickness is positive, and in the opposite direction when thickness is negative; the offset being half the thickness.

Examples

ExpdSurf1.ms2 - The basis surface is just a flat rectangle, so the strain solution is bypassed.

ExpdSurf2.ms2 - The basis surface is already flat, so the strain solution is bypassed.

ExpdSurf3.ms2 - The basis surface is developable but not flat (a cone). Constraint would have no effect, since the strain is 0.

ExpdSurf4.ms2 - The basis surface is part of a sphere. Changing constraint to +1 has no effect, since the strains are already all negative. Changing constraint to -1 causes the flattened surface to expand; a uniform positive strain of approximately 10% has been applied to make all strains nonnegative. Edges come out about 10% longer.

ExpdSurf5.ms2 - The basis surface is a Translation Surface with one u=constant breakline. The Expanded Surface gets a degree-1 breakline in the same place.

If you move point p2 to Y = 0, the Expanded Surface straightens out, and its breakline should really disappear -- there is no discontinuity left, of any degree. However, the program isn't smart enough to figure that out, at this time. The Expanded Surface currently gets a breakline wherever the basis surface has one, with the same degree.

ExpdSurf6.ms2 - Same as ExpdSurf5.ms2, except the supports of the Translation Surface have been reversed, so now it's a v=constant breakline.

ExpdSurf7.ms2 - flattens part of a sphere. Two Expanded Surfaces are made. s1 has constraint = 0; you can check the lengths of the Edge Snakes in Mass Properties and see they are accurately preserved. s2 has constraint = -1, contraction only. You'd need to do a lot of contraction of the outer portions of s2, especially the boundary, to turn it into the curved s0.

ExpdSurf8.ms2 - is a more complex example with 2 u-breaks and 1 v-break. It doesn't immediately look like it's flat, because of the wavy u and v parameter lines. But, rotate the view a little so your eye is in the plane of the Expanded Surface.

See also: Expanded TriMesh

Expanded TriMesh (Flattener authorization required)

Parents and characteristic properties

Degree = Sub-division amount
Kind = expansion plane
Surface/TriMesh = the basis surface
Point0 = used to locate and orient the expansion in the plane of flattening
Point1 = used to orient the expansion in the plane of flattening
Thickness = thicknes of the material for the surface
Frame = location of expanded surface
Constraint = -1 or 0 or +1
Scale1 = scale factor for primary axis stretch
Scale2 = scale factor for secondary axis stretch

Description

The Expanded TriMesh is a surface-class entity that encapsulates conformal flattening of a parametric surface. This entity allows flat pattern development to take place entirely within the environment of SurfaceWorks, rather than saving a file and performing flattening with a separate program.

The Surface/TriMesh specifies the surface to expand. It is expanded as follows: Point0 and Point1 are either projected onto Surface/TriMesh, or even better are magnets. Point0 is then placed at the origin of Frame and the surface is expanded so that Point1 lies along the Frame's axis as specified by Kind.

Kind

x-y plane(default) the Expanded Surface lies in the x,y plane of Frame, with the image of Magnet along Frame's x-axis y-z plane the Expanded Surface lies in the y,z plane of Frame, with the image of Magnet along Frame's y-axis

z-x plane the Expanded Surface lies in the z,x plane of Frame, with the image of Magnet along Frame's z-axis.

Surface/TriMesh the basis surface (an offset from it, actually, at half the thickness) is the surface to be flattened.

Point0 and Point1 are used to orient the flat pattern in the plane of expansion. The flattened surface is rotated so the image of Point0 is at the origin of Frame and Surface/TriMesh is then expanded with Point1 in the positive axis direction as specified by Kind. If Point0 and Point1 are points, then they are projected onto Surface/TriMesh. For this reason, you will likely achieve more predictable results if you make Point0 and Point1 Magnets or TriMesh Magnets depending on your choice of Surface/TriMesh.

Constraint is an option, similar to MSPlex and Flattener, to apply an overall strain adjustment to the flattened surface, to compensate for limitations of the forming process.

constraint = -1: strain is constrained to be negative (contraction), in going from flat to curved
constraint = 0: no strain constraint. All edge lengths are accurately preserved
constraint = +1: strain is constrained to be positive (expansion), in going from flat to curved.

Scale1 and Scale2 are applied at the end. They just multiply the coordinates in the primary and secondary axis directions, respectively. One motivation for scale factors is fabric stretch factors in apparel and signage applications. Another is an observation that some users like to have a little safety margin; it's relatively easy to trim a plate that comes out a little large after forming, but it's a much bigger problem when one comes out slightly small.

A triangle mesh (roughly 2000 triangles) is made from the surface, and flattened by the trimesh flattening algorithm.

The strain distribution is calculated and saved internally, but is not currently visible to the user except through Strain Contours, Strain command, and STRAIN function.

(If the basis surface is already flat, the strain solution is bypassed; strain is identically zero, and constraint and thickness have no effect.)

Thickness is applied, similar to our other flattening methods. The actual surface triangulated for flattening is a uniform normal offset of the basis surface, in the direction of the basis surface positive normal when thickness is positive, and in the opposite direction when thickness is negative; the offset being half the thickness.

Examples

ExpdTriMesh0.ms2 - This is a very small, simple example with only 4 triangles, in a pyramid configuration. Two Expanded TriMeshes are made from it.

I0 uses * for its frame, and its kind is 1, so it lies in the global X,Y plane, with the image of G0 at the origin, and the image of G1 along the X axis. Because constraint is 0, and both stretch factors are 1.00, the boundary strain is 0; in <z> view you can see that the perimeter is identical to the original.

I1 is similar, but uses F0 for its frame.

ExpdTriMesh1.ms2 - Similar to ExpdTriMesh0.ms2, with these differences:

1. TriMesh tm has subdivision degree 1, therefore 16 triangles. Its two corners at p1 and p2 are breakpoints; these corners stay sharp.
2. l0 has a Scale2 of 1.5
3. l1 has a subdivision degree of 1, resulting in 64triangles

ExpdTriMesh2.ms2 - This starts with a more complex TriMesh (96 triangles), with a breakline across it.

The strain is pretty dramatic, -42%.

See also: Expanded Surface

Foil Lofted Surfaces

Parents and characteristic properties

Type = type of airfoil family, 1 to 5; types 101 - 255 reserved for user-defined foils
Master curves = master curves (t-orientation doesn't matter)

Description

In the Foil Lofted Surface, the lofting curves are Foil Curves. This means the Foil Lofted Surface passes through its first (trailing edge), third (leading edge), and fourth or fifth (trailing edge; if present) master curves, but in general not the others, which control the thickness and camber much as the second and fourth (if present) control points of a Foil Curve do.

Similarly to a Foil Curve, the number of master curves (3, 4, or 5) controls whether you generate a half or full profile.

Half-section surface

With 3 master curves, you get one half of a symmetric foil-lofted surface, with the parameter v running from 0 at the trailing edge to 1 at the leading edge. The second curve controls the half-thickness of the symmetric foil; this is the distance of the second curve from the ruled surface consisting of all the chord lines joining the leading and trailing edges.

Full-section surface

With 4 or 5 master curves, you get a full-section foil-lofted surface — that is, if the first curve and the last curve coincide, or are the same curve. When the first and last curves are not the same, you still get a more-or-less foil-shaped surface, but the trailing edge is open.

5 master curves. This creates a full section surface:

symmetric — when curve2 and curve4 are symmetrically placed (mirror images) with respect to the chord line

cambered — when curve2 and curve4 are NOT symmetrically placed with respect to the chord line curve2 and curve4 act together in determining the thickness and camber. The full-section foil will be cambered if curve2 and curve4 are not symmetrically placed with respect to the chord line.

4 master curves. This creates a symmetric full section. A 4-point foil behaves just like a 5-point foil in which the 4th parent is the mirror image of the 2nd parent across the chord line.

u,v parameters. The u parameter on the surface is the same as the t parameter for the first master curve. The v parameter is the parameter along the lofting foils, running from 0 at the trailing edge (upper surface) to 0.5 at the leading edge, then to 1.0 at the trailing edge (lower surface). The lofting foils are the u = constant lines.

Type

type-1 NACA 4-digit series; max camber at 30%

type-2 NACA 63 series with a=0.3 mean line

type-3 NACA 64 series with a=0.4 mean line

type-4 NACA 65 series with a=0.5 mean line

type-5 NACA 0010-34; max camber at 40%

types 101-255 user-defined foils

relabel is used to control the labeling of the v-direction (the lofting curves) of the surface. See Reference section “Relabeling Curves and Snakes.”

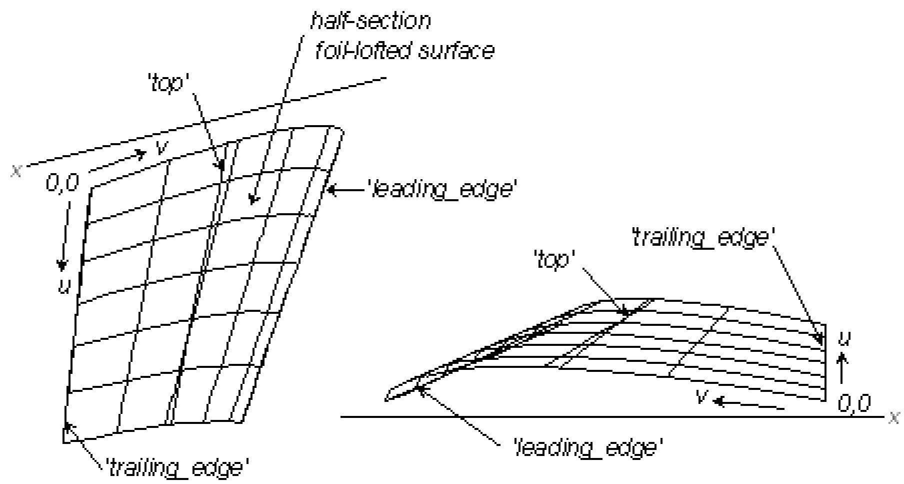
Usage

The Foil Lofted Surface can be used to create complex wing, keel, rudder, and hydrofoil geometries having true NACA sections or user-defined foil sections.

Example 1: floft1.ms2 (half-section)

Example 1: floft1.ms2 (half-section)

‘keel’ is a type-1 half-section foil-lofted surface defined by 3 master curves: ‘trailing_edge’, ‘top’, and ‘leading_edge’. ‘keel’ passes through ‘trailing_edge’ and ‘leading_edge’ and uses ‘top’ to establish the half-thickness of the foil surface.



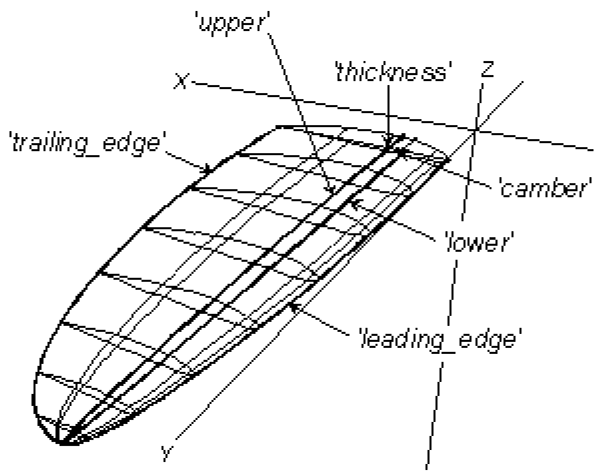
FLOFT1.MS2 at Lat -30, Lon 70 (left).

Example 2 floft2.ms2 (full-section)

Example 2: floft2.ms2 (full-section)

FLOFT2.MS2 is a cambered elliptic wing made from 5 master curves that are full foil sections: ‘trailing_edge’, ‘upper’, ‘leading_edge’, ‘lower’, and ‘trailing_edge’ (again). Note that ‘trailing_edge’ is named twice in order to close the surface. In the model (but not the figure below), tickmarks are turned on for the master curves so you can see them more easily.

Camber and thickness can be adjusted independently, by sliding RelBeads ‘camber’ (cyan) and ‘thickness’ (red) respectively. (The model is constructed so that the t-offset of RelBead ‘camber’ is the camber / chord ratio, and the t-offset of RelBead ‘thickness’ is half the thickness / chord ratio.)



FLOFT2.MS2
at Lat 30, Lon 60
(tickmarks turned off).

See also: Foil Curve, Foil snake

Foil Curve

Parents and characteristic properties

Type = type of airfoil family, 1 to 5; types 101 - 255 reserved for user-defined foils
Defining Points = point entity. control points

Description

A Foil Curve is a true NACA airfoil section. Foils can be constructed with 3, 4, or 5 control points — 3 for a half section, 4 for a symmetric full section, or 5 for a cambered or symmetric full section. In all cases:

- The first Defining Point is the trailing edges.
- The third Defining Point is the leading edge.
- The first and third Defining Points (trailing and leading edges) form the “chord line”.
- The second and fourth Defining Points (top and bottom camber) of a 5-point foil, determine thickness and camber.
- The last control point in a 4- or 5-point foil (second trailing edge) nominally closes the foil. This is usually the same point as the first Defining Point (trailing edge).

For details about defining half-section and full-section foils, see "Understanding SurfaceWorks Entities - Collective Entity Information - Foil Curves".

type selects a standard NACA foil family or a user-defined foil:

type-1 NACA 4-digit series; max camber at 30%

type-2 NACA 63 series with a=0.3 mean line

type-3 NACA 64 series with a=0.4 mean line

type-4 NACA 65 series with a=0.5 mean line

type-5 NACA 0010-34; max camber at 40%

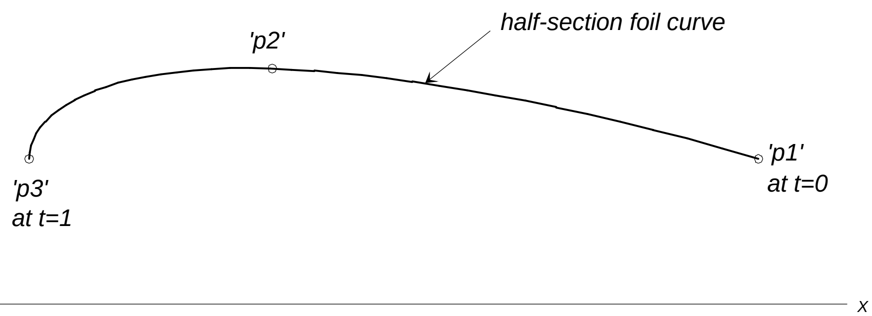
types 101-255 user-defined foils

For details about foil types (standard and user-defined), see "Understanding SurfaceWorks Entities - Collective Entity Information - Foil Curves".

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is concentrated in the highly curved leading edge area. Subdivision control may be important if you are creating panels for a potential-flow code.

Example1: foil1.ms2

'profile' is a type-1 half-section foil curve formed from the three points 'p1', 'p2', and 'p3'. Since all the points are at Y = 0, the curve lies in the Y = 0 plane. 'p1' (the trailing edge) and 'p3' (the leading edge) form the "chord line". The distance of 'p2' from the chord line (3.25 - 2 = 1.25 units) establishes the half-thickness.

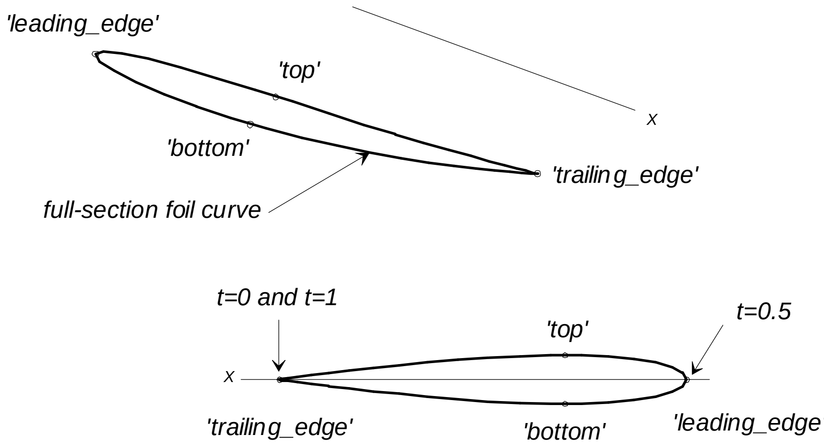


Foil1.

Example2: foil2.ms2

'root' is a type-1 full-section foil curve formed by 5 points: 'trailing_edge', 'top', 'leading_edge', 'bottom', and 'trailing_edge' (again). If you look at the list of control points for 'root' in the Edit>Definition dialog box, you can see that 'trailing_edge' is both "Point defining trailing edge" and "Point defining second trailing edge". Choosing the points in this way will durably close the section.

This foil curve does not lie in a plane. 'trailing_edge' and 'leading_edge' form the chord line and both 'top' and 'bottom' determine the thickness of the section. Since 'top' and 'bottom' are symmetrically spaced on either side of the chord line, the foil curve has no camber.



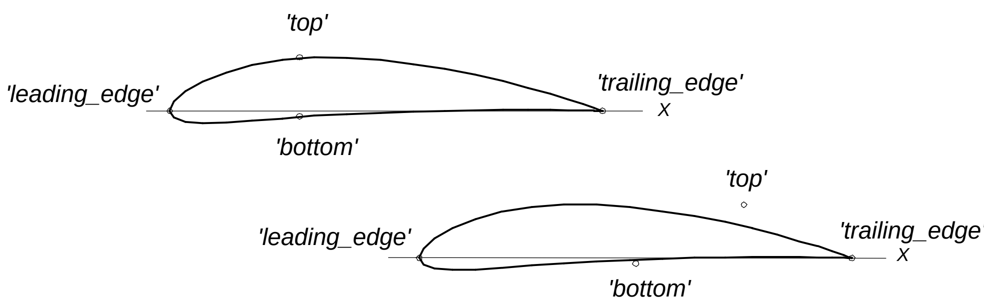
Foil2.

Example3: foil3.ms2

The full-section foil curve in this example differs from the curve just above in only 2 ways:

‘top’ is at Y = .5
‘bottom’ is at Y = -.05

Since these 2 points are no longer symmetrically distant from the chord line, this foil is cambered (top figure).

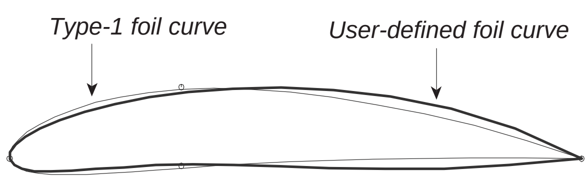


Foil3 in Home view.

The bottom figure shows what happens to the foil curve if the fore-and-aft positions of its second and fourth points (‘top’ and ‘bottom’) are changed. Yes, this shift does not change the curve at all!

Example4: foil7 (needs type165.foi)

The curve 'userdef' is a user defined foil curve made from the same points as 'root', the type-1 foil curve used in the first part of Example 3. Type for 'userdef' is 165. Being a type greater than 100, this number tells SurfaceWorks to look for the file TYPE165.FOI which contains the information specifying the user-defined foil shape.



Foil Snake

Parents and characteristic properties

magnet1 ... magnetN = control points

Description

type = type of airfoil family, 1 to 5; types 101 - 255 reserved for user-defined foils

plane. This maps into a foil shape lying in the surface its 3 or 5 control point magnets are attached to. If the net of u-v parameter lines in the vicinity is roughly rectangular, the Foil Snake will be close to a true NACA section or user-defined foil in space.

Foils can be constructed with either 3, 4, or 5 control magnets — 3 for a half section, 4 for a symmetric full section, or 5 for a cambered or symmetric full section. In all cases:

- The snake lies in the same surface as its magnets.
- magnet1 is the trailing edge.
- magnet3 is the leading edge.
- The “chord line” is the Line Snake joining magnet1 and magnet3.
- point2, and point4 of a 5-point foil, determine thickness and camber.
- The last control point in a 4- or 5-point foil nominally closes the foil (usually same point as point1).

type:

type-1 NACA 4-digit series; max camber at 30%

type-2 NACA 63 series with a=0.3 mean line

type-3 NACA 64 series with a=0.4 mean line

type-4 NACA 65 series with a=0.5 mean line

type-5 NACA 0010-34; max camber at 40%

types 101-255 user-defined foils

relabel is used to relabel the snake. The default relabel ‘*’ produces a labeling in which the parameter t is concentrated in the highly curved leading edge area, in the u-v parameter space. Subdivision control may be important if you are creating panels for a potential-flow code.

Notice

The implementation of FSnake may be changed in a future version. We have found FSnakes to be of marginal value because of the distortion of the foil shape that occurs when the true foil outline is expressed in u, v parameters and mapped onto the distorted mesh of u, v grid lines. We now see the possibility of compensating for this distortion, producing a much truer rendition of the foil.

The ProjSnake entity provides a way to overcome this problem: instead of using an FSnake, use a ProjSnake made from an FCurve. For example, in the FSNAKE2.MS2 example below, you could make an FCurve lying in a horizontal plane such as Z = 0, then project it vertically onto the hull surface.

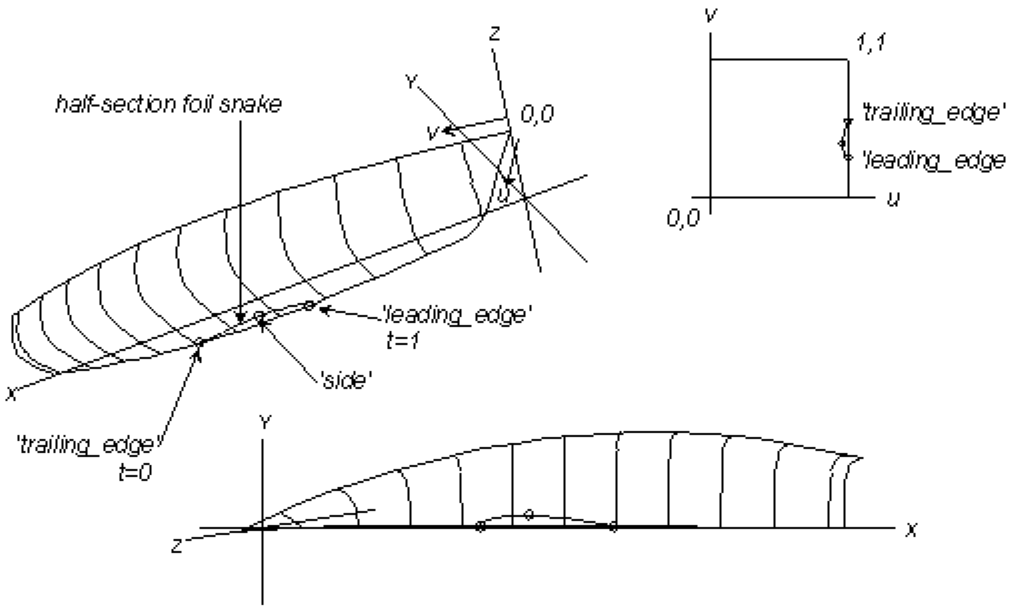
Example fsnake2.ms2

Example: fsnake2.ms2

‘hull’ is a C-lofted surface made from 5 master curves ‘MC1’ to ‘MC5’. u = 0 is the sheerline, and v = 0 is ‘MC1’.

‘trailing_edge’, ‘side’, and ‘leading_edge’ are three magnets on ‘hull’. ‘trailing_edge’ and ‘leading_edge’ are located on the u = 1 edge of the surface.

‘keel_root’ is a type-1 half-section foil curve, defined by the three magnets and lying on ‘hull’.



FSNAKE2.MS2 at Lat -30, Lon 50 (top) and Lat 90, Lon -90 (bottom).

If ‘hull’ is changed by modification of one or more of the master curves, ‘keel_root’ will change its position in space but will remain always on ‘hull’, in a corresponding position.

See also: Foil curve

Formula

Constants and real entities

There are two ways to specify a real number in the syntactic definition of an object: with a constant value or a real entity. Variables and Formulas are example of real entities. A Variable carries a value that lies within a specified range. A formula involves an expression made of real entities, constants, operators and functions.

Constants

Examples of constants are 1.23, -0.7, 12e-8, +98, etc. Although they have no explicit units, they are always considered to carry the correct units needed at the particular location where they are used.

For instance in

Point P 1 1 / 1 2 3 ;

1, 2 and 3 are considered as having length units.

In

Magnet M 1 1 / S 0.123 0.456 ;

the u and v parameter values 0.123 and 0.456 are considered to be unitless.

Variables

Please see "" on page 442.

Formulas

A formula expresses a computation from constants, variables and other reals with operators and functions; it carries units resulting from its expression.

Basic syntax

The simplest syntax for a formula is:

Formula name / { expression } ;

For instance

Formula c / { 177.69 } ;

Formula f / { x } ;

Formula s / { x + y } ;

Formula s / { x + 5 * y } ;

Expression

Operators

The expression of a formula can be made with the usual operators '+' (addition), '-' (subtraction), '*' (multiplication), '/' division. Exponentiation is symbolized with '^' and is evaluated right to left, i.e. x^y^z is x^(y^z) and not (x^y)^z. Parentheses can be used to prioritize the evaluation.

Formula d / { b^2 – 4*a*c } ;

Formula P / { R*I^2 } ;

Formula V / { n * R * T / P } ;

Formula s / { (1-t^2) / (1+t^2) } ;

Functions

A set of mathematical functions and object accessors is available:

Name	Argument(s)	Result	Synopsis
ABS	1, any units	Same units as argument	Absolute value
ACOS	1: unitless	unitless	arc cosine (radians)
ACOSD	1: unitless	unitless	arc cosine (degrees)
ALARM	2: any units	Unitless	ALARM has 2 arguments ALARM(x,y). The alarm "goes off" (goes into error) if (1) it is set (x > 0) AND (2) y < 0. Using a formula or expression for y, you can build various warning limits into a model.
ANGLE	3: point, point, point	Unitless (degree)	Angle of three points (angle at pt2 between the directions to pt1 and pt3)
ARCLEN	3: curve, unitless, unitless	Length	Arc distance along curve, from t1 to t2
AREA	2: surface, use_sym (0 or 1)	Area = L^2	Area of surface, CompSurf, or TriMesh
ASIN	1: unitless	unitless	arc sine (radians)
ASIND	1: unitless	unitless	arc sine (degrees)
ATN	1, unitless	Radian (unitless)	Arc tangent
ATND	1, unitless	Degree (unitless)	Arc tangent (in degrees)
ATN2	2, both with same units	Radian (unitless)	Arc tangent(y/x)
ATN2D	2, both with same	Degree (unitle	Arc tangent(y/x) (in degrees)

	units	ss)	
BBOX	1. Entity or Entity List 2. Real scale factor 3. Real sign 4. Index, 1 to 3 for X, Y, or Z component	Length	The BBOX function gets information about the bounding box of an entity, or a set of entities specified by an Entity List. A bounding box is the smallest rectangular solid, aligned with the global coordinate system, that encloses the selected entities.
BSPL	1. KnotList, or *UNIFORM for uniformly spaced knots. 2. K, polynomial order (2 for linear, 3 for quadratic, 4 for cubic, etc.) 3. N, number of basis functions. 4. I, index indicating which basis function to evaluate (1 to N). 5. T, parameter (nominal range 0 to 1, but can be any real value)	unitless	The BSPL function evaluates the so-called "B-spline basis functions", which are the mathematical foundations of B-spline and NURBS curves and surfaces. Example: BSPL(*UNIFORM, 3, 5, 2, 0.40) returns 0.3200 In this case the knots are uniform (0, 0, 0, 1/3, 2/3, 1, 1, 1); the B-splines are quadratic (K = 3); there are N = 5 of them; I = 2 selects the second basis function; T is 0.40. Errors: 222. NURB has too few knots for its order and number of control points. 223. NURB has too many knots for its order and number of control points. 234. Insufficient spacing between knots. 556. BSPL function: order less than 1. 557. BSPL function: number of basis functions less than 1. 558. BSPL function: index is out of range (1 to number of basis functions).

CEIL	1: any units	Same units as argument	CEIL(x) is the smallest integer that is greater than or equal to x
CENTROID	3: entity, use_sym (0 or 1), index (1-3, for X,Y,Z coordinate)	Length	Coordinates of centroid
CLEAR	2: point, graphic entity	Length	Clearance
COS	1, radian (unitless)	Unitless	Cosine
COSD	1, degree (unitless)	Unitless	Cosine (of angle in degrees)
COSH	1: unitless	unitless	hyperbolic cosine
CURV		1/Length	Curvature of host curve or snake, at t location of bead/ring. If t is on a breakpoint, hi_side (0 or 1) controls whether curvature is measured below or above the break. kind: 0 is 3-D curvature of curve or snake; 1 is normal curvature of snake; 2 is geodesic curvature of snake.
CURVINT	3: curve, t, real	L times units of real	integral of real times ds along curve ds is the element of arc length along the curve t is a Variable real is a Formula descended from t
DIST	2: point, point	Length	Distance between points
ERROR	1: entity	Unitless	Error code attached to entity (0 if no error).
EXP	1, unitless	Unitless	Exponential
FLOOR	1: any units	Same units	FLOOR(x) is the greatest integer that is less than or

		as argument	equal to x
FRAMEPOS	3: point, frame, index (1-3, for x,y,z coordinate)	Length	Coordinates of point in frame
GRAPH	2: graph, unitless	Unitless	Evaluation of graph
HYDRO	6: sp.gr., Zcg, sink, trim, heel, index	various, depending on index	fixed-position hydrostatics based on the visible contours index is 1 to 29; selects one of 29 results, e.g. index = 6 for displacement volume; index = 15 for wetted surface
IF	3: any units	Same as units of selected argument	If arg1 >0, arg2; else arg3
LOG	1, unitless	Unitless	Natural logarithm
LOG10	1, unitless	Unitless	Base-10 logarithm
MASS	3: entity, use_sym, index	M ML	Mass, if use_sym is not 0, includes symmetry images Index = 0 returns Mass Index = 1, 2 or 3, the value returned is the mass moment with respect to X, Y or Z. This is the product of mass times the X, Y or Z coordinate of the centroid. Unit dimensions are ML
MAX	2, both with same any units	Same units as arguments	Maximum
MIN	2, both with same any units	Same units as arguments	Minimum
PI	1; any units	Unitless	PI has 1 argument, but its value is immaterial; PI(x)

			= pi for any x
ROUND	1, any units	Same units as argument	Rounding to integer
ROUND2	1, any units	Same units as argument	(x, places) rounds x to the specified number of decimal places. E.g., ROUND2(PI(0),2) is 3.140000.
SIGN	1: any units	Unitless	SIGN(x) is +1 when x > 0, -1 when x < 0, 0 when x = 0
SIN	1, radian (unitless)	Unitless	Sine
SIND	1, degree (unitless)	Unitless	Sine (of angle in degrees)
SINH	1: unitless	unitless	hyperbolic sine
SQRT	1, unit dimensions all multiples of 2	Unit dimensions of argument divided by 2.	Square root
STRAIN	2: Surface/TriMesh, index	Unitless	Surface/TriMesh is a surface or TriMesh entity index = 0 or 1, for minimum or maximum strain This function reports the strain range for an Expanded Surface or Expanded TriMesh.
SURFCURV	5: magnet, hi_side_u, hi_side_v, kind, angle	L^-1 for kind = 0 or 2; L^-2 for kind = 1	Surface curvature kind = 0, normal curvature kind = 1, Gaussian curvature kind = 2, mean curvature
SURFINT	4: surface, u, v, real	L^2 times units	integral of real times dA over surface

		of real	dA is the element of area on the surface u and v are Variables real is a Formula descended from u and v
TAN	1, radian (unitless)	Unitless	Tangent
TAND	1, degree (unitless)	Unitless	Tangent (of angle in degrees)
TANH	1: unitless	unitless	hyperbolic tangent
TPOS	1, bead or ring	Unitless	t parameter
UNITMASS	1: entity	M for a point ML^-1 for a curve ML^-2 for a surface ML^-3 for a solid	unit weight property of entity
UPOS	1, magnet or ring	Unitless	u parameter
VELOCITY	3: curve, t, hi_side	Length	Rate of change of arc length with respect to t If t is on a breakpoint, hi_side (0 or 1) controls whether velocity is measured below or above the break.
VOLUME	2: solid, use_sym (0 or 1)	Volume = L^3	Volume of solid
VPOS	1, magnet or ring	Unitless	v parameter
XPOS	1, point	Length	X coordinate
YPOS	1, point	Length	Y coordinate
ZPOS	1, point	Length	Z coordinate

Examples:

Formula l / { A * SIN(omega * t + phi) } ;

Formula x / { EXP(COS(phi)) } ;

Formula y / { SQRT(ABS(b^2 - 4 * a * c)) } ;

Formula z / { TAN(c * ANGLE(p1, p2, p3) + phi^2) / b } ;

Rules

Unit consistency

Real entities cannot appear just anywhere inside a formula: unit consistency must be observed. The units of the result of a formula must be specified, and must match the units generated by the expression. The rules are:

1 Only reals of same units can be added or subtracted.

2 Exponentiation of an expression with a rational exponent (this includes the SQRT function) is only possible if the result has integral unit dimensions, e.g. SQRT(x^2 + y^2) is valid; (x^3)^(1/4) is not, if x and y have units of length.

3 Constants are considered to have the same units as the other operand in a sum and no units in a product, e.g. .in 1 – x, 1 is considered to have the same units as x. This is offered as a facility: no physically meaningful formula has unit-less additive constants, since the formula would become inconsistent in case of unit scaling.

4 Function arguments and results have unit dimensions as specified in the preceding table.

Unit mismatch is an error. Unit consistency is checked at parse time when possible (the unit dimensions of x^y cannot be determined without knowing y's value). It is however completely checked at evaluation time.

No simplification

Expressions are not simplified, therefore

```
Formula f / { x - x } ;
```

depends upon x (twice!), which gets evaluated when f needs to be.

No optimization

No optimization is performed which would prevent parents from being evaluated, e.g. in

```
Formula f / { x * y } ;
```

y will be evaluated even if x is 0.

Complete syntax

'Formula' name { attr, } { 'R:1' } ' / ' '{' expr '}' { units } ';'

Due to the fact that numeric constants are needed in formulas, and that MS2 syntax allows entity names such as 12, 4e3, etc., entitiies with such names will not be recognized inside the expression of a formula, between { and }; these symbols will be considered as numeric constants instead. Likewise, entities having the same name as a function (SIN, SQRT, XPOS...) will not be recognized.

Examples

```
Point V1 1 1 / 1 2 3 ;
Point V2 1 1 / 4 5 6 ;
Formula V1dotV2 / { XPOS(V1)*XPOS(V2)
                    +YPOS(V1)*YPOS(V2)
                    +ZPOS(V1)*ZPOS(V2)
                  }L^2;
Variable Intensity  / 2.5 I (0,10) ;
Variable Resistance / 100 L^2 M T^-3 I^-2(0,500) ;
Formula  DPotential / { Resistance * Intensity^2 } M T^-3 I^-2 ;
```

Real entities as parents

Real entities can of course be parents of Variables and Formulas. But they can be parents of all entitiies depending on a real value, and wherever an entity regardless of its type can be a parent, such as in an entity list. Therefore they can be parents of points (as coordinates, offsets, angles, t-, u- and v-parameter), curves and surfaces (via knot lists and weights), etc.

```
Variable A / 10 L ;
Variable B /  5 L ;
Variable C / 30 L ;
Variable D / 25 L ;
Point P1  14  1 / 0 0 0 ;
Point P2  14  1 / A 0 0 ;
Point P3  14  1 / A B B ;
Point P4  14  1 / C B B ;
Point P5  14  1 / C 0 0 ;
Point P6  14  1 / D 0 0 ;
Variable k0 / 0.25 ;
Variable k1 / 0.75 ;
KnotList kl / { 0 k0 0.5 k1 1.0 } ;
Variable w0 / 1 ;
Variable w1 / 3 ;
NURBCurve spline0 11 5 64x1 / * 2 kl { P1 1 P2 w0 P3 w1 P4 1 P5 1 P6
w0 } ;
```

Note that since a Variable is defined by three real numbers, these can be replaced by real entities: you can for instance have a Variable with a variable range:

```
Variable min / 0 ;
Variable max / 24 ;
Variable x   / 7 (min,max) ;
```

In this case the entities used as value and range bounds must all carry the same units (as always, constants are assigned the units that match).

User interface

Figure 2 shows the properties for the newly created Formula ‘ArcRadius’. Note the Unit dimensions are specified as “Length”. Figure 3 is the Expressions dialog. The

DIST function was chosen from the “Functions” drop-down and the two points were added from the “Entities” drop-down. The expression was completed, by adding the closed parenthesis from the dialog, or your keyboard.

When you are editing a real value in the data for any entity, you can use an appropriately dimensioned real entity instead of a constant. For example, creating or editing a RadiusArc, one required data element is the Radius. When you select this item a constant or a real entity can be entered. Figure 4 shows a Radius Arc with the Radius field active and a radius of 1. To enter a Formula, go to the Surfer View and choose the appropriate Formula. (Figure 5) Figure 6 shows Formula ‘ArcRadius’ as a parent, which has a value of 0.500.

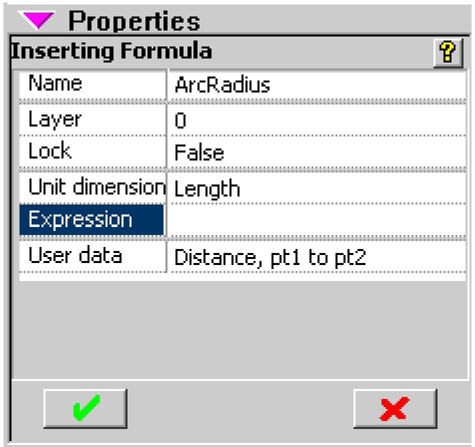


Fig 2

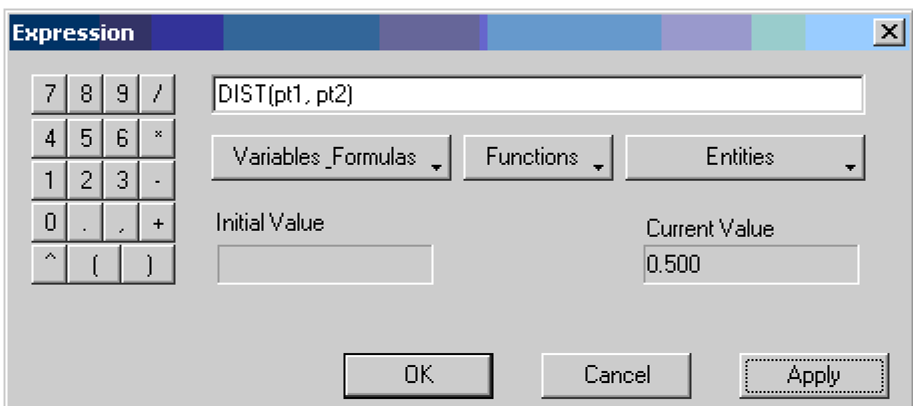


Fig 3

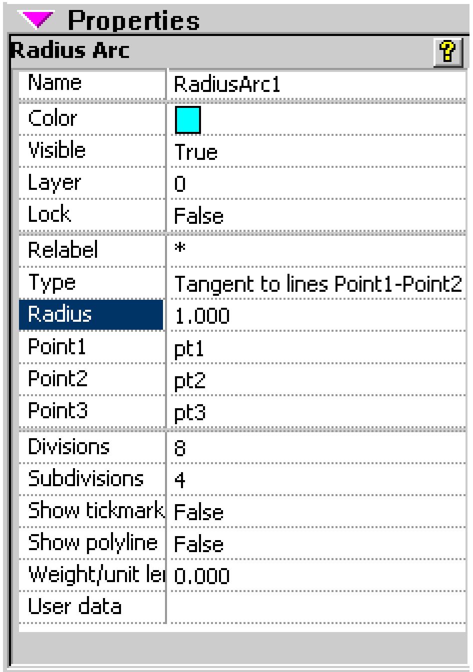


Fig 4

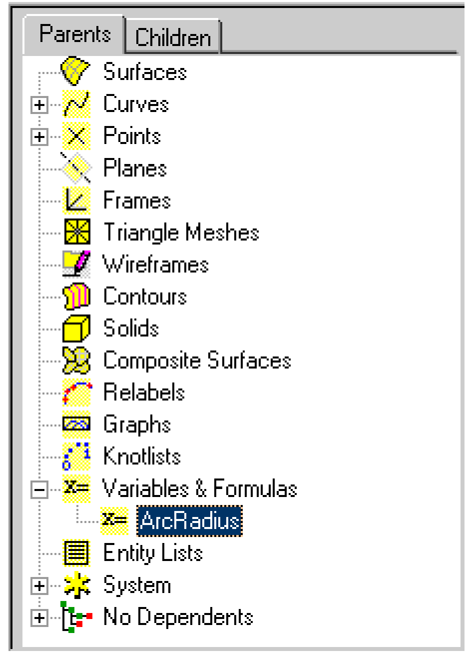


Fig 5

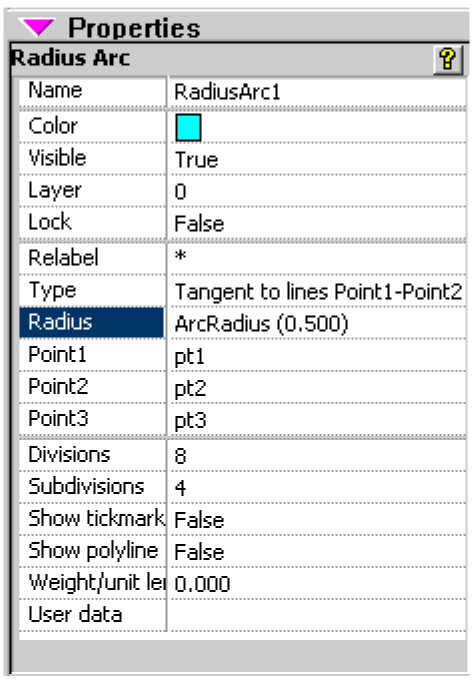


Fig 6

Example Application <Install Directory>\Examples\IBeam.ms2

The following syntax generates the profile of an I-beam given four parameters: Height, Width, FlgThk, WebThk (note how FlgThk, WebThk are upper-bounded).

Variable ***Height*** / 100.000 L (10.000, 200) ;

Variable ***Width*** / 70.000 L (10.000, 150) ;

Variable ***FlngThk*** "flange thickness" / 20.000 L (0.000, ***Height***) ;

Variable ***WebThk*** / 20.000 L (0.000, ***Width***) ;

Formula ***d1*** / { (***Width*** + ***WebThk***) / 2 } L ;

Formula ***d2*** / { ***Height*** - ***FlngThk*** } L ;

Formula ***d3*** / { (***Width*** - ***WebThk***) / 2 } L ;

FrameAbsPt BotFlgLLC 14 1 / * 0.000 0.000 0.000 ;

FrameAbsPt BotFlgLRC 14 1 / * ***Width*** 0.000 0.000 ;

FrameAbsPt BotFlgURC 14 1 / * ***Width FlngThk*** 0.000 ;

FrameAbsPt WebLRC 14 1 / * ***d1 FlngThk*** 0.000 ;

FrameAbsPt WebURC 14 1 / * ***d1 d2*** 0.000 ;

FrameAbsPt TopFlgLRC 14 1 / * ***Width d2*** 0.000 ;

FrameAbsPt TopFlgURC 14 1 / * ***Width Height*** 0.000 ;

FrameAbsPt TopFlgULC 14 1 / * 0.000 ***Height*** 0.000 ;

FrameAbsPt TopFlgLLC 14 1 / * 0.000 ***d2*** 0.000 ;

FrameAbsPt WebULC 14 1 / * ***d3 d2*** 0.000 ;

FrameAbsPt WebLLC 14 1 / * ***d3 FlngThk*** 0.000 ;

FrameAbsPt BotFlgULC 14 1 / * 0.000 ***FlngThk*** 0.000 ;

BCurve I_profile 11 1 12x1 / * 1

Geodesic Snake

Parents and characteristic properties

curvature = a geodesic curvature value (defined below)
curvature graph = name of a B-spline Graph object or '*' for default graph
log-tolerance = logarithm (base 10) of tolerance for matching prescribed curvature
magnet1 locates the t=0 end of the snake
magnetN locates the t=1 end of the snake

Description

A Geodesic Snake is a curve embedded in a surface, with a specified distribution of geodesic curvature along its length (for definition of geodesic curvature, see "Geodesic Curvature" below). The geodesic curvature can be zero (constant) in which case the Geodesic Snake is a geodesic on the host surface.

Note that calculation of a Geodesic Snake is a computationally intensive searching process, without guarantee of success (details below in "Failure to converge"). Therefore, unless your task really requires accurate control over geodesic curvature, we recommend that you use a simpler snake entity, such as a Line Snake or B-spline Snake.

A Geodesic Snake is defined by 2 or more magnets which all must lie in the same surface. The program uses a type-2 B-spline Snake made from the magnets as its starting path for computing the Geodesic Snake. If you give only 2 magnets, start and end, the type-2 B-spline Snake is just the Line Snake from magnet1 to magnet2. For many Geodesic Snake applications, especially those with zero curvature, two magnets will be sufficient. If the resulting Geodesic Snake fails to converge, you can add one or more intermediate magnets to get a better starting configuration.

A Geodesic Snake can have zero (constant) geodesic curvature for its entire length, or it can have a specified distribution of geodesic curvature. curvature and graph are the controls:

curvature is a geodesic curvature value, units = 1 / length. 0 curvature makes an actual geodesic satisfying Geodesic conditions (1), (2) below.

graph modulates the geodesic curvature; i.e., multiplies the curvature value. The default graph '*' is interpreted as a constant, 1.

log-tolerance allows you to specify a tolerance for the convergence of the iterative search SurfaceWorks uses to compute the Geodesic Snake. The value is specified as the log (base 10) of tolerance (using a logarithm here allows a wide dynamic range). Example: log-tolerance = -4 means tolerance = 10^{-4} = .0001. log-tolerance is provided as a control because practical accuracy requirements for geodesics vary widely between applications.

Tolerance itself is the largest tolerable deviation of calculated geodesic curvature from specified geodesic curvature, multiplied by the total arc length of the Geodesic Snake (this makes it non-dimensional). Example: a curve with 10-meter arc length and maximum curvature of 10^{-3} / meter (curvature x arc length = 10^{-4}) could deviate from a straight line by no more than 0.125 mm.

relabel is used to relabel the snake. The default relabel '*' produces the "natural" relabeling of the snake which is uniform with respect to arc length.

Geodesic curvature

Geodesic curvature is a pointwise scalar property of a curve lying in a surface, determined as follows:

1. At any point along the curve, construct the plane tangent to the surface.
2. Project the local portion of the curve normally onto the tangent plane.
3. Geodesic curvature is the curvature of the projection.

Geodesic

A geodesic is a curve embedded in a surface, with a special property which can be expressed in either of two mathematically equivalent ways:

1. A geodesic has zero geodesic curvature throughout its length.
2. A geodesic is the shortest curve in the surface joining its endpoints.

Failure to converge

The typical way a Geodesic Snake fails to evaluate is "failure to converge." Features which can promote this problem are:

- coarse subdivisions of the host surface

- irregular mesh on the host surface
- lack of smoothness of the host surface
- passing through or close to a coordinate singularity
- starting path far from the geodesic path
- tight tolerance

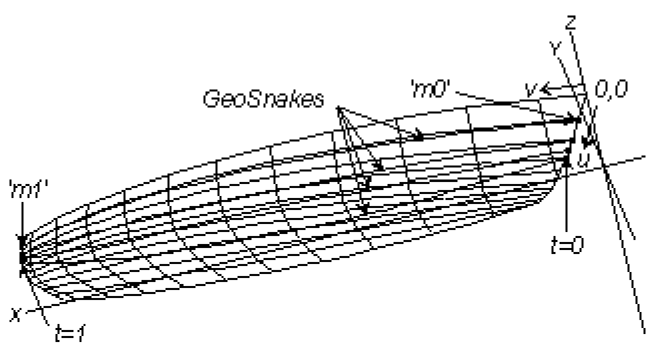
Usage

Geodesics have significant applications in manufacturing:

1. Ribbands — they must lie along geodesic paths if they are to conform to the hull surface between frames (see “Example”).
2. Planked wooden boat construction — geodesic seams produce straight planks.
3. Cold-molded plywood construction — veneers have to lie along geodesic paths. (Take a look at VENEERS.MS2 which shows a hull in the process of veneer layout. After making the first veneer-edge Geodesic Snake, we saved it as a component, then loaded it successively, moving each new Geodesic Snake to its proper location by moving its Ring control. Then we’d make Developable Surfaces (the veneers) between adjacent pairs of Geodesic Snake.)
4. Sail making — geodesic seams are key to efficient use of cloth and flat panel development. (Take a look at SAILSEAM.MS2 which shows a jib in the process of being seamed; eventually, Developable Surfaces would be created between each adjacent pair of seams.)
5. Plate expansion — a mesh of geodesic triangles is key step in the process.
6. Compounding by line heating — using geodesics for heat lines minimizes residual stress.
7. Automated lamination — reinforcing tape has to be laid along geodesic paths, or it wrinkles and buckles.

Example: ribbands.ms2

Example: ribbands.ms2



In this example, Geodesic Snakes are used to layout the paths for ribbands. The use of geodesics makes ribbands that will naturally lie flat on the frames and conform to the hull surface between frames.

'm0' is an Magnet on the v = 0 edge of the hull; 'm1' is a Magnet on the v = 1 edge of the hull (at du = 0, dv = 1 from 'm0'). 'n0' is the Geodesic Snake between 'm0' and 'm1'.

To make the set of ribbands, we saved this first ribband as a component (objects 'm0', 'm1', and 'n0'; RIBBAND.MC2), then loaded the component successively to make the other ribbands. Since we made the initial ribband so that it can be moved by moving just the magnet 'm0', it is easy to move each component ribband to its own position on the hull.

Note that where the hull has relatively high curvature at the turn of the bilge, we had to play with the position of the control magnet to get the geodesic we wanted. In this region, a small movement up or down of 'm0' makes a big difference in the geodesic path.

See also: LineSnake

Helix

Parents and characteristic properties

Point = point entity. start of helix
Axis line = Line. axis for the helix
Pitch = slope
Angle = total angle = 360 x number of turns

Description

This curve entity produces a Helix starting at Point (t=0) and using Axis line as its axis.

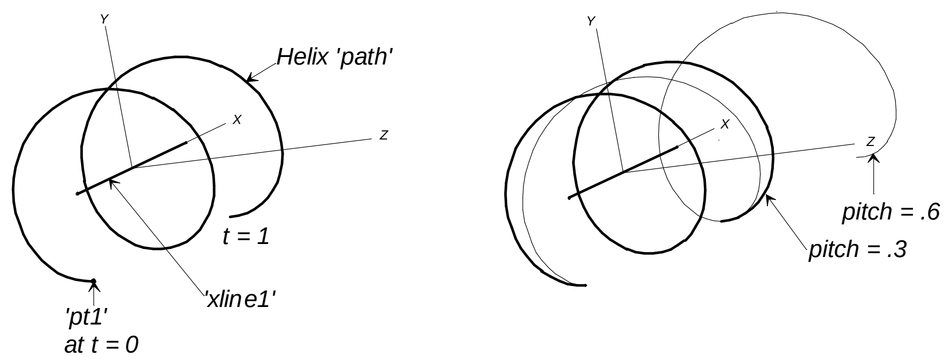
Pitch is a real number controlling the slope of the Helix with respect to the rulings on the surface of the cylinder the Helix lies on (i.e., the cylinder having Axis line as its axis and passing through Point). The Helix advances a distance of 2 × pi × radius × Pitch in making one full turn. Positive Pitch makes a right-handed helix, e.g. a standard right-handed screw thread. If Pitch = 0, the Helix is a circular arc.

Angle is the total angle of the helix in degrees, e.g. 360 for one full turn, 720 for two turns, etc.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is uniformly distributed along the length of the curve.

Example: Helix-Sweep4.ms2

'path' (cyan) is a Helix beginning at 'pt1' (black) and having 'xline1' (magenta) as its axis line. The Helix is used to give the hose a coiled shape. The pitch of 'path' is 0.3 and its total angle is 720 degrees.



Helix-Sweep4 showing the Helix ‘path’ and its parents: initial helix (left); with pitch doubled (right).

The right-hand figure shows ‘path’ (Pitch = 0.3) with a second helix ‘helix2’ (red; hidden in the model) made from the same data except that Pitch = 0.6. To show this helix, expand the "Curves" section of the Surface Manager, right click on 'helix2' and choose "Show" from the drop down menu.

The hose surface is a Sweep Surface using the following parents:

- the Bead ‘bead1’ (green; at the t=0 end of ‘path’); designates the Helix as the path curve for the surface
- the Arc ‘shape’ (bright magenta; to see it, zoom in on the area around ‘pt1’)
- the Relative Curve ‘guide’ (blue)

Hydrostatic Real Entities

Parents and characteristic properties

t = specific gravity
f = Zcg
f = sink
d = trim
d = heel
T = kind (index)
(x) = one or more Contours entities

Description

The HydroReal is provided to do real time hydrostatics in MultiSurf.. Implementing these enities are best accomplished by loading the provided component, Hydro.MC2. This component is found in the Examples folder. The only parent nereded to load the component is an Entity List, which contins all the transverse contours intended to be used for Hydrostatics.

Please reference the table below for the description of each Index.

Index	Units of Result	Result

1	Length	forward end of waterline
2	Length	aft end of waterline
3	Length	waterline length
4	Length	waterline maximum beam
5	Length	maximum draft
6	Length^3	displaced volume
7	Length	X coordinate of center of buoyancy
8	Unitless	center of buoyancy as % of waterline
9	MLT^-2	displacement
10	Length	Y coordinate of center of buoyancy
11	Length	Z coordinate of center of buoyancy
12	Length^2	waterplane area
13	Length	X coordinate of center of flotation
14	Unitless	center of flotation as % of waterline
15	Length^2	wetted surface
16	Length	X-coordinate of wetted surface center
17	Length	Z-coordinate of wetted surface center
18	Length	transverse metacentric height
19	ML^2T^-2	transverse righting moment per degree
20	Length	longitudinal metacentric height
21	ML^2T^-2	longitudinal righting moment per degree
22	Unitless	waterplane coefficient
23	Unitless	prismatic coefficient
24	Unitless	block coefficient
25	Unitless	maximum section coefficient

26	Unitless	displacement-length ratio
27	Length^2	lateral plane area
28	Length	X coordinate of lateral plane center
29	Length	Z coordinate of lateral plane center

Intersection Bead

Parents and characteristic properties

Type = curve / curve or curve / surface
bead/curve = curve to be intersected (object being cut)
mirror/surface = basis cutting plane or surface
bead/curve1 = first cuve to intersect (Curve / Curve type only)
bead/curve2 = second cuve to intersect (Curve / Curve type only)
Tolerance Intersection tolerance (Curve / Curve type only)
point = specifies the actual cutting plane or surface

Description

An Intersection Bead is a point on a curve, located where a cutting surface intersects the curve.

The object being cut is designated by bead/curve (either a bead or a curve):

When bead/curve is a <u>bead</u>, it serves two purposes:

1. It specifies which curve the IntBead is to lie on: the curve that already supports bead.

2. In case of multiple intersections of the curve with the cutting surface, the IntBead will be located at the intersection nearest to bead.

When bead/curve is a <u>curve</u>, it specifies directly which curve the IntBead is to lie on. In case of multiple intersections, the one closest to t = 0.5 will be taken.

The cutting object is designated as a combination of a mirror/surface and a point:

mirror/surface defines an infinite family of <u>potential</u> cutting surfaces. It can be either a mirror (plane, Line, or point) object or a surface object.

point specifies which one of the infinite family of cutting surfaces is the actual cutting surface. point specifies the <u>distance</u> from mirror/surface to the cutting surface. When point lies on mirror/surface, the distance is zero. point can be any point object.

The following table shows the effect of the various choices for cutters:

mirror/surface	**cutting surface (a construction surface; not a real object)**

plane (mirror)	the parallel plane which passes through point — point specifies the perpendicular distance from the specified plane to the cutting plane
line (mirror)	the circular cylinder which has its axis along this line and which passes through point — point specifies the radius of the cylinder
point (mirror)	the sphere which is centered at the mirror and passes through point — point specifies the radius of the sphere
surface (surface)	the parallel surface which passes through point — point specifies the perpendicular distance from the specified surface to the cutting surface When mirror/surface is a surface, usually the cutting surface you want is the surface itself, not one of its parallel surfaces. In this case, you can use any magnet on the surface for point; this assures a zero offset.

Example: Piston.ms2

Example: piston.ms2

This model, the skeleton of a reciprocating engine, shows an example of the use of an IntBead to control the length of a moving part in a mechanism.

Bead ‘e1’ (green) is at the intersection of the crankshaft axis and the cylinder axis. RelPoint ‘tdc’ (magenta) is in a fixed position above ‘e1’, establishing the eccentric radius of the crank (0.35).

Bead ‘e5’ is the center of the crankpin. RelPoint ‘p5’ is a fixed distance (1.25) above Bead ‘e5’; this distance establishes the length of the connecting rod (which runs from Bead ‘e5’ to IntBead ‘e2’).

IntBead ‘e2’ (blue) lies on the cylinder axis and uses both ‘p3’ and ‘p5’:

the object being cut is the cylinder axis

the cutting object is designated by the combination of:

- mirror = Bead ‘e5’— since this is a point object, the infinite family of potential cutting surfaces are <u>spheres</u>, centered at ‘e5’

- point = RelPoint ‘p5’ — it designates the actual cutting sphere, out of the infinite family of potentials; in this case, since RelPoint ‘p5’ is 1.25 units from ‘p3’, the cutting sphere is the one with radius = 1.25 (this is the length of the connecting rod)

You can vary the t-location of Bead ‘e5’ to crank the engine

Example: IntBead2-1.ms2, Intbead2-2.ms2

Intersection Magnet

Parents and characteristic properties

magnet/surface = surface to be intersected (object being cut) bead/curve = cutting curve

Description

An Intersection Magnet is a point on a surface, located where a curve (extended if necessary) intersects the surface.

The object being cut is designated by magnet/surface (a magnet, ring, or surface):

When magnet/surface is a magnet, it serves two purposes:

1. It specifies which surface the IntMagnet lies on: the surface that supports magnet.

2. In case the line intersects the surface more than once, the IntMagnet will be located at the intersection nearest to magnet.

When magnet/surface is a surface, it specifies directly which surface the IntMagnet is to lie on. In case of multiple intersections, the one closest to u = 0.5, v = 0.5 will be taken.

The cutting object is designated by bead/curve (either a bead or a curve):

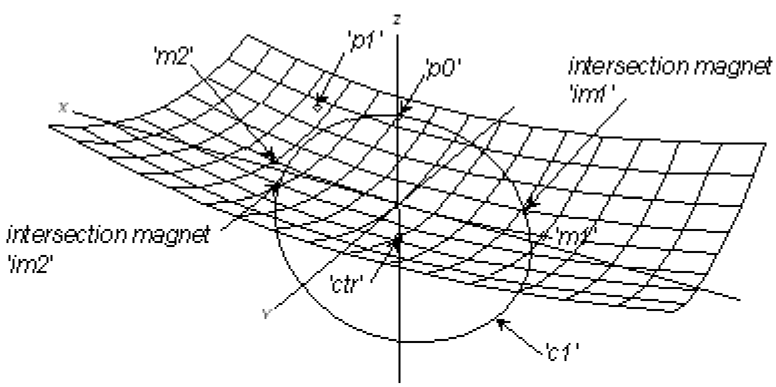
When bead/curve is a bead, the cutting object is the curve the bead belongs to.

When bead/curve is a curve, it specifies the cutting object directly.

Example: Intersection Magnet.MS2

This example uses IntMagnets to locate the two intersections of a surface and a circle.

The TranSurf ‘patch’ (cyan) and the circle ‘c1’ (green) intersect at two points. AbsMagnets ‘m1’ and ‘m2’ (magenta), located in the vicinity of the two intersections, are used to give the program good starting places from which to search for the intersections for each of the IntMagnets ‘im1’ and ‘im2’ (red).



Intersection Magnet.MS2 at Lat 30, Lon 60.

See also: Intersection Bead, Intersection Ring, Intersection Snake

Intersection Point

An Intersection Point is located at the common intersection point of three planes and / or surfaces. (Actually, it solves a considerably more general intersection problem, as it allows offsets to be specified from any combination of the 3 planes and / or surfaces.)

Parents and characteristic properties

Magnet/surface/plane/frame1
Magnet/surface/plane/frame2
Magnet/surface/plane/frame3
Offset1, Offset2, Offset3

Description

If a magnet is specified in place of a surface, the magnet location is used as a starting point in the search for an intersection point.

As usual, a frame can serve as a plane; in this case, the plane is the x-y plane of the frame.

When the intersector is a surface, the offset is positive in the direction of the positive normal (like an OffsetSurf). Bear in mind that such an offset will be reversed if the surface's orientation attribute is reversed.

Planes have an orientation, too, though this has not so far been made a visible characteristic of a plane, so you need to follow some rules if you want to make use of an offset from a plane:

*X=0, *Y=0, *Z=0 -- A positive offset is in the positive coordinate direction.

Plane2 -- A positive offset is in the direction from Point1 toward Point2

Plane3 -- A right-hand rule prevails; if you made a Frame3 from the same three points, picked in the same order, positive offset is in the direction of the positive z-axis.

OffsetPlane -- Orientation is the same as that of the parent plane.

Frame -- A positive offset is in the direction of the positive z-axis.

Example

Intersection Point1.ms2. This has 4 IntPoints;:

ip0 is at the intersection of 3 planes.

ip1 is at the intersection of 2 planes and 1 surface.

ip2 is at the intersection of 1 plane and 2 surfaces.

ip3 is at the intersection of 3 surfaces.

Intersection Ring

Parents and Characteristic Properties

Intersection Type = Snake-Surface
Host Ring/Snake = snake (or ring on snake) to be intersected (entity being cut). ring will belong to this snake.
Mirror/Surface = plane, surface, Line, or point entity. basis cutting plane or surface

Point = point entity. specifies the actual cutting surface or plane

Intersection Type = Snake-Snake
Host Ring/Snake = snake (or ring on snake) to be intersected (entity being cut). ring will belong to this snake.
Intersecting Ring/Snake = the cutting snake or ring on cutting snake

Description

A Snake-Surface Intersection Ring is a point on a snake, located where a cutting surface intersects the snake.

A Snake-Snake Intersection Ring is a ring at the intersection of two snakes on the same surface. The Intersection Ring in this case belongs to the Host Ring / Snake; it is located where the Intersecting Ring / Snake crosses Host Ring / Snake. Since the Snake-Snake Intersection Ring is a ring, it can serve as a magnet on the surface that the two snakes belong to.

The entity being cut for a ***Snake-Surface Intersection Ring*** is designated by Host Ring/Snake (either a ring or a snake):

When Host Ring/Snake is a ring, it serves two purposes:

1 It specifies which snake the Intersection Ring is to lie on: the snake that is the parent of Host Ring.

2 In case of multiple intersections of the snake with the cutting surface, the Intersection Ring will be located at the intersection nearest to Host Ring.

Note: When using a ring for the Host Ring/Snake, it is important to always check the resultant location of the Intersection Ring. Because of the way SurfaceWorks finds the intersection, there can sometimes be unusual results. It is best to place the ring quite close to the desired intersection. See "Understanding Entities - Collective Entity Information - Intersection Entities".

When Host Ring/Snake is a snake, it specifies directly which snake the Intersection Ring is to lie on. In case of multiple intersections, the one nearest t = 0.5 will be taken.

The entity being cut for a ***Snake-Snake Intersection Ring*** is designated by Host Ring/Snake (either a ring or a snake):

When Host Ring/Snake is a ring, the snake being cut is the snake that Host Ring belongs to, and the t-value of Host Ring is used as the starting t on the snake being cut.

When Host Ring/Snake is a snake, it identifies the snake being cut directly; the starting t on the snake being cut will be 0.5.

The cutting surface for a ***Snake-Surface Intersection Ring***, in order to allow cutting flexibility, is specified as a combination of a Mirror/Surface and a Point:

The cutting object is designated as a combination of a Mirror/Surface and a Point:

Mirror/Surface can be either a mirror (plane, line, or point) or a surface object.
Point can be any point object.
Mirror/Surface defines an infinite family of potential cutting surfaces; Point specifies which one of this infinite family is to do the cutting.

The following table shows the effect of the various choices for a cutting object:

Mirror/Surface	**Cutting Surface (a construction surface; not a real entity)**
plane (Mirror)	the parallel plane passing through Point — Point specifies the perpendicular distance from the specified plane to the cutting plane
line (Mirror)	the circular cylinder with its axis along this line and which passes through Point — Point specifies the radius of the cylinder
point (Mirror)	the sphere centered at the mirror that passes through Point — Point specifies the radius of the sphere
surface (Surface)	the offset surface that passes through Point — Point specifies the perpendicular distance from the specified surface to the cutting surface

The cutting snake for a ***Snake-Snake Intersection Ring***, is designated by Intersecting Ring/Snake (either a ring or a snake):

When Intersecting Ring/Snake is a ring, Intersecting Ring/Snake is the snake that the ring belongs to, and the t-value of the ring is used as the starting t on Intersecting Ring/Snake.

When Intersecting Ring/Snake is a snake, it identifies Intersecting Ring/Snake directly; the starting t on Intersecting Ring/Snake will be 0.5.

SurfaceWorks performs an iterative search for the crossing of the two snakes. If there is only one intersection, occurring reasonably near the middle of each snake, and the snake shapes are simple, it is usually sufficient just to name the snakes. In more complicated cases, it is valuable to name a ring, both to specify which of two or more intersections you want, and to help the program locate the intersection.

Host Ring/Snake and Intersecting Ring/Snake can be the same; this will search for the intersection of the snake with itself (if any). In this case, you must use rings and put them near the intersection, one on each “end” of the snake (see Example "intring3").

Note that although the specification of the two snakes is quite symmetric, and the resulting ring does in principle lie on both snakes, the Intersection Ring belongs only to Host Ring/Snake. If you should need a ring on Intersecting Ring/Snake as well, at this intersection, you’d have to make another Intersection Ring, with the appropriate parents.

For more details about specifying intersection entities, see "Understanding Entities - Collective Entity Information - Intersection Entities".

Example: IntRing.ms2

This model is an example of an Intersection Ring with multiple intersections.

‘n1’ is a B-spline Snake (green) lying on the Translation Surface ‘patch’. ‘a1’ is a 2-point Plane (yellow) which intersects ‘n1’ three times. ‘r1’ (white) is a Ring which serves to (1) tell which snake the Intersection Ring is to belong to, and (2) give the program a good starting place from which to search for the intersection to be used for the location of the Intersection Ring ‘ir’ (magenta).

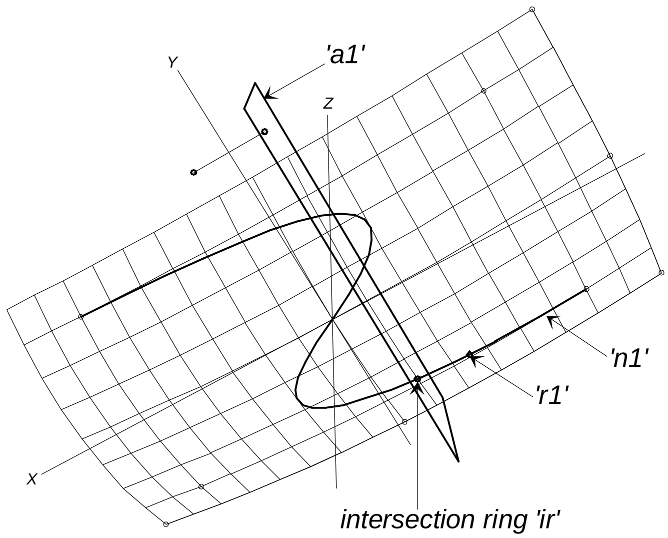


Fig. 1. IntRing.ms2 (initial model).

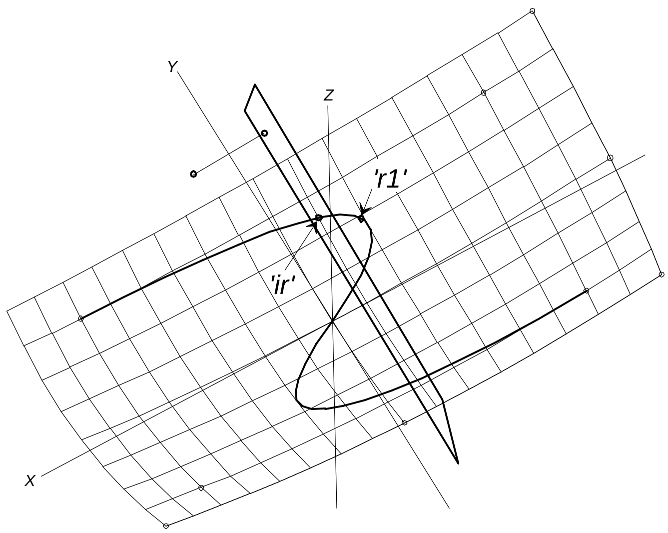


Fig. 2. IntRing.ms2 with ‘r1’ moved to t = .7

You can slide ‘r1’ to a new location, to see how this affects the position of ‘ir’ (Fig. 2). You can also change the Host Ring/Snake to ‘n1’; this will give the program a good start for finding the intersection near the middle of ‘n1’, the same as positioning ‘r1’ at t = 0.5.

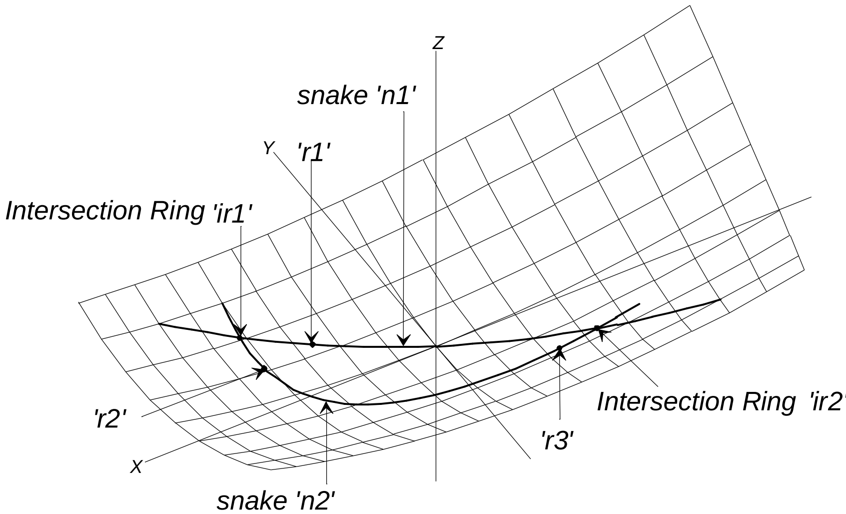
Example: IntRing2.ms2

In this example, there are two intersections between Line Snake ‘n1’ (magenta) and C-spline Snake ‘n2’ (yellow). An Intersection Ring is made at each intersection:

‘ir1’ (white) has the rings ‘r1’ and ‘r2’ (both green) for parents. Therefore, ‘ir1’ belongs to snake ‘n1’ (because the parent ring ‘r1’ belongs to snake ’n1’), and it lies at the crossing of snake ‘n2’.

‘ir2’ (cyan) has ring ‘r3’ and snake ‘n1’ for parents. Therefore, ‘ir2’ belongs to snake ‘n2’ (because the parent ring ‘r2’ belongs to snake ‘n2’), and it lies at the crossing of snake ‘n1’.

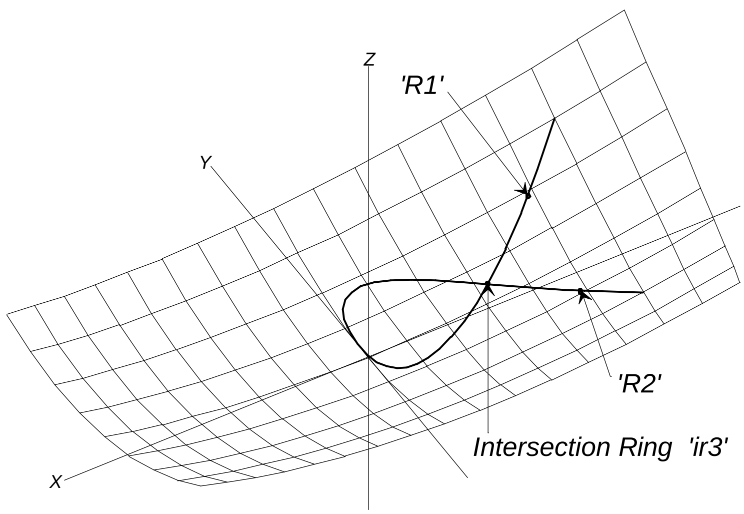
In each case the correct crossing is selected by positioning the seed rings ‘r1’, ‘r2’, ‘r3’ reasonably close to the desired intersection.



IntRing2.ms2.

Example: Intring3.ms2

This example shows an Intersection Ring ‘ir3’ (blue) located at the crossing of B-spline Snake ‘n0’ with itself. ‘ir3’ has ‘R1’ and ‘R2’ for parents. The supporting rings are placed close to the intersection and on the two tailing ends of the snake, to make clear instructions for the program.



IntRing3.ms2.

See also: Intersection Snake

Intersection Snake

Parents and characteristic properties

Magnet = magnet or ring. specifies the surface being cut
Mirror/Surface = plane, surface, Line or point. defines a family of potential cutting planes or surfaces

Point = point entity. specifies the actual cutting plane or surface

Description

An Intersection Snake is a curve on a surface, located where a real or imaginary cutting plane or surface intersects the surface.

The basic data for an Intersection Snake are two surfaces that logically can intersect. We think of the first of these entities as being cut and the other as doing the cutting or being the cutter. The surface doing the cutting can be the actual Mirror/Surface entity (if Point lies on that surface), or it can be an imaginary cutting surface (for details, see "***The cutting surface***..." below. The entity resulting from the intersection is a snake on the surface being cut.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, which is proportional to arc length in the u-v parameter space.

The entity being cut is designated by Magnet; it serves three purposes:

1. It specifies which surface the Intersection Snake is to lie on: the surface that is parent to Magnet.
2. In case of multiple intersections of the surface with the cutting surface, the Intersection Snake will be made from the piece nearest to Magnet.
3. The t=0 end of the Intersection Snake will be the end that is nearest to Magnet.

The cutting surface, in order to allow cutting flexibility, is specified as a combination of a Mirror/Surface and a Point:

Mirror/Surface defines an infinite family of potential cutting surfaces. It can be either a mirror (plane, Line, or point entity) or a surface.

Point specifies which one of the infinite family of cutting surfaces is the actual cutting surface. Point can be any point entity and specifies the distance from Mirror/Surface to the cutting surface. When Point lies on Mirror/Surface, the distance is zero.

This combination defines the cutting surface, which is not a real SurfaceWorks entity; it is only an imaginary construction surface.

The following table shows the effect of the various choices for cutting surfaces:

Mirror/Surface	**Cutting Surface (a construction surface; not a real entity)**
plane (Mirror)	the parallel plane passing through Point — Point specifies the perpendicular distance from the specified plane to the cutting plane
line (Mirror)	the circular cylinder with its axis along this line and which passes through Point — Point specifies the radius of the cylinder
point (Mirror)	the sphere centered at the mirror that passes through Point — Point specifies the radius of the sphere
surface (Surface)	the offset surface that passes through Point — Point specifies the perpendicular distance from the specified surface to the cutting surface

When Mirror/Surface is a surface, usually the cutting surface you want is the surface itself, not one of its parallel surfaces. In this case, you can use any magnet on the surface for Point; this assures a zero offset.

Tips

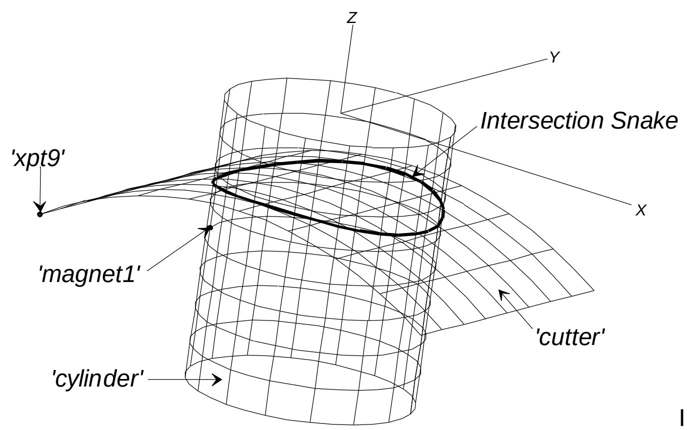
Alternate construction: When making an Intersection Snake (which will lie on the surface being cut), avoid using a surface for the entity doing the cutting when possible. Instead, consider using a plane (cutting surface is flat) or a line (cutting surface is a cylinder) because they evaluate 10 to 20 times faster. Also consider alternate constructions such as a Projected Snake.

Intersections: In all cases of calculation of intersections, there is the possibility that the desired intersection is in fact nonexistent. Just because a surface and a cutting surface might intersect doesn't mean they actually do intersect. Don't worry — non-existent intersections are trapped as geometry errors.

At the same time, there is often the possibility that two entities will intersect in two or more places. SurfaceWorks handles this by giving you the option to "point" to the correct intersection, whichever is closer to Magnet on the surface being cut.

There is another awkward possibility — that the intersection actually has the wrong dimensionality. Normally, the intersection of a surface and a cutting surface would be a curve. It could be a surface, however, if the two happen to coincide over some area. In general, these higher-dimensional, coincidence type intersections will produce non-useful results or geometry errors.

Example: IntSnake.ms2



IntSnake

'cylinder' (green) is a B-spline Lofted Surface that we are cutting with the Translation Surface 'cutter' (blue). To find the intersection of the two surfaces, we made the Intersection Snake 'intsnake' (magenta) using the following parents:

Magnet = 'magnet1' (red). Since this magnet lies on 'cylinder', it identifies the cylinder as the parent of the Intersection Snake.

Mirror/Surface = 'cutter'. This identifies the infinite family of surfaces parallel to 'cutter' as potential cutting surfaces.

Point = 'xpt9' (dark magenta). In the family of potential cutting surfaces, this identifies the Translation Surface itself as the cutting surface.

See also: Intersection Ring, Projected Snake

Knot List

Parents and characteristic properties

Knots = a list of knots

Description

Knot List provides an optionally non-uniform knot set for construction of NUB-spline Fitted and NURBS entities. It is not a visible object in itself, so it has no color or visibility.

The list of values must:

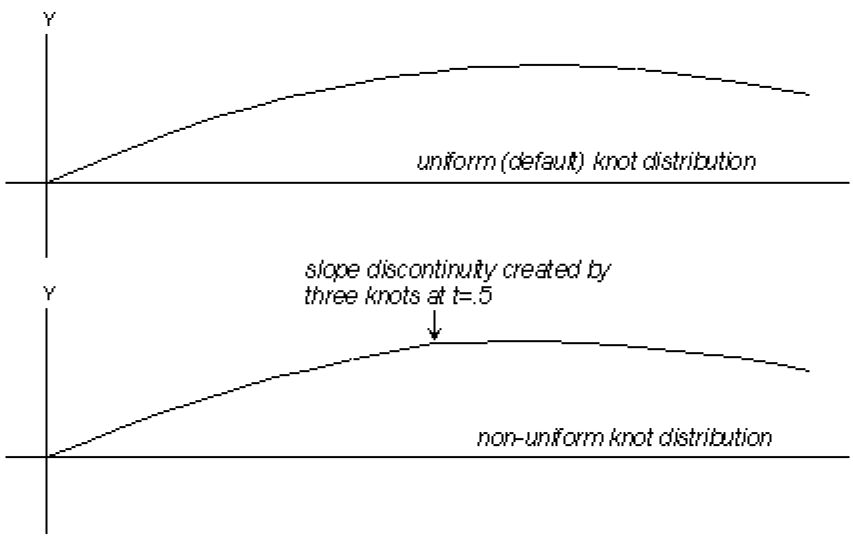
be composed of non-decreasing numbers

begin with 0 and end with 1

contain a quantity of values that varies according to the number of control points and the Degree of NUB-spline Fitted or NURBS entity involved. The Knot List should have (number of control points) - (Degree) + 1 members. For example, if the number of control points = 6 and Degree = 3, you'd need 6 - 3 + 1 = 4 members; { 0 .25 .75 1 } would be a possible set of values. (See Examples).

Drawing knots together allows the curve to have high curvature in the vicinity of the multiple knots.

Examples



KNOTLST0.MS2 initial model (top); KNOTLST1.ms2, edited (bottom).

In this example (upper figure), 'curve' (a NURBS Curve) uses the default '*' for its knotlist attribute, telling SurfaceWorks to use the default knot distribution for the curve. This distributes the knots evenly along 'curve'.

In the lower figure, 'knots1' is a Knot List object used to make non-uniform knots for 'curve' (the same NURBS Curve as above, but now with different knot distribution). Since the NURBS Curve has 7 control points and is type-3, the KnotList needs 7 - 3 +

1 = 5 values, including the beginning 0 and the ending 1. The set of knot values { 0. .5 .5 .5 1. } puts triple knots at t = .5, creating a slope discontinuity (a bend) in the curve at t = .5.

See also: NUB-spline Fitted Curve, NUB-spline Fitted Snake, NUB-spline Fitted Surface, NURBS Curve, NURBS Snake, NURBS Surface

Light TriMesh

Parents and characteristic properties

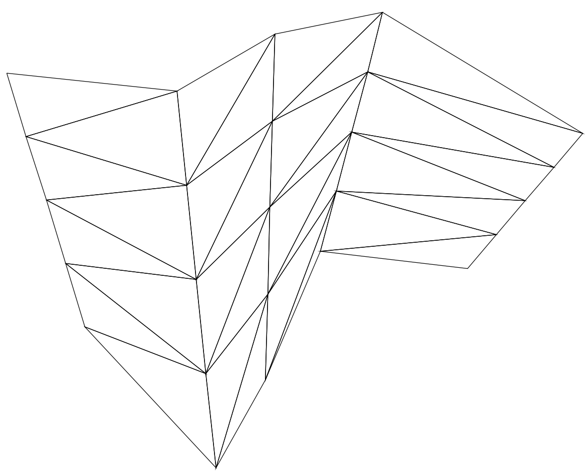
Frame = frame for the parent points

Description

The Light TriMesh, exists to support TriMesh imports from RAW and STL files. It is supported in SurfaceWorks in that it can be imported using **File>Import>Triangular mesh** and to a limited extent edited.

The points that are used to create the Light TriMesh in SurfaceWorks are all related to a frame. The choice of that frame can be edited. (The rest of the geometric data cannot be edited).

Example: trimesh2.ms2



trimesh2 is an example of a Light TriMesh.

Line

Parents and characteristic properties

Endpoints = point entity. point1, point2

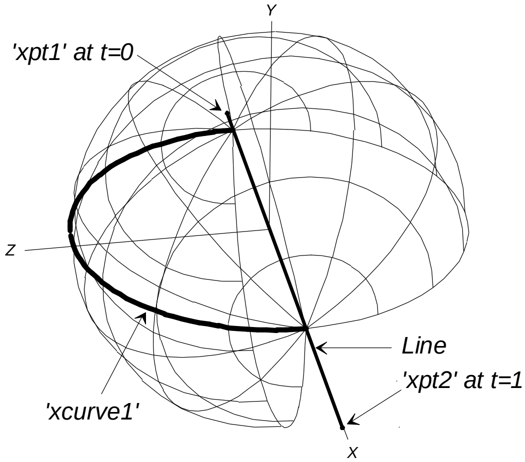
Description

A Line is the simplest type of curve. It is just a straight line joining point1 (where t = 0) to point2 (where t = 1). point1 and point2 can be any point entities.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t's distribution is "linear" or "uniform" along the line.

Example: RevSurf1-Line.ms2

Line ‘axis’ (brown) between endpoints ‘xpt1’ and ‘xpt2’ is used as the axis for Revolution Surface ‘surface1’.



RevSurf1-Line

The arc ‘xcurve1’ is revolved around ‘axis’ from -180 degrees to 90 degrees, forming three-quarters of a sphere.

See also: Line Snake

Line Snake

Parents and characteristic properties

End magnets = magnet or ring. magnet1, magnet2

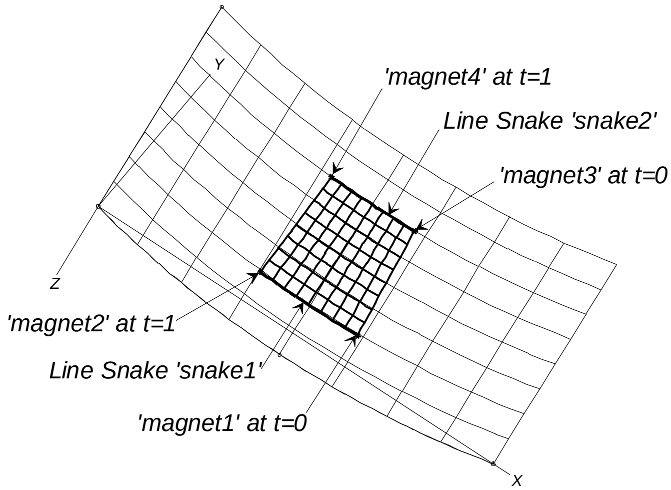
Description

A Line Snake is the simplest type of snake. It is just a straight line joining magnet1 (where t = 0) to magnet2 (where t = 1) in the u-v parameter space. It is constrained to lie on a surface, spanning the distance between 2 magnets.

In many circumstances, a Line Snake is approximately geodesic, i.e. close to the shortest snake joining the two magnets. However, this is only approximate, and on highly curved surfaces, or surfaces on which the parameter lines have a lot of curvature, the geodesic curvature of a Line Snake can be considerable.

Relabel is used to relabel the curve. The program's default Relabel ‘*’ produces the "natural" labeling, in which the parameter t's distribution is "linear" or "uniform" along the snake, in the u-v parameter space.

Example: LineSnake.ms2



LineSnake

In the above example Line Snake 'snake1' was created using parent magnets 'magnet1' and 'magnet2' on the surface 'sweep' (green). Line Snake 'snake2' was created using parents 'magnet3' and 'magnet4' from the same surface. These two snakes are in turn parents to SubSurface 'patch' (blue).

See also: Line

Magnet

Parents and characteristic properties

Surface = the parent surface for the magnet
u and v (or du and dv when Relative to is checked) = location
Relative to (when checked) = the parent magnet for relative location

Description

A Magnet is located by specifying the Surface and the absolute parameter values u, v (or du and dv offset from another magnet on the same surface). A Magnet can be used wherever a point or magnet is called for. Magnets are especially useful for defining snakes.

Surface can be any kind of surface. When Relative to is checked, the Surface field will gray out — the magnet you choose to offset from will suffice to identify the surface on which the relative Magnet will lie.

The default location for a Magnet depends on the contents of the Selection Set when the magnet is created:

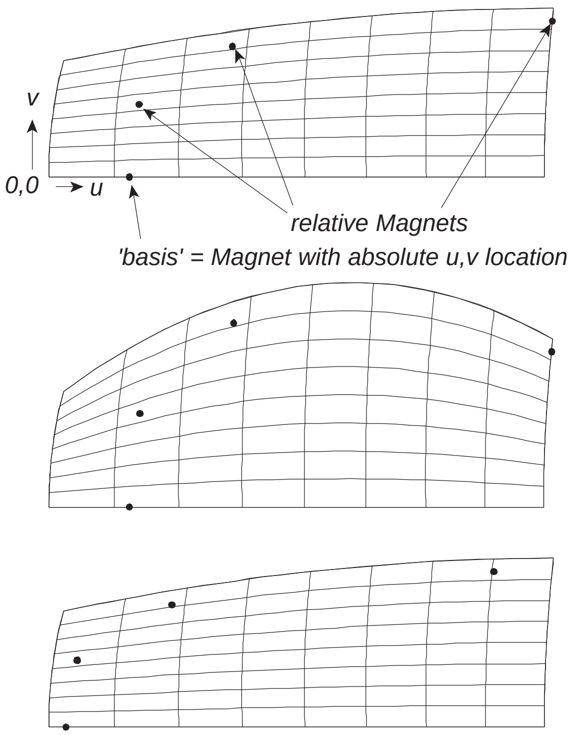
When the Selection Set consists of a single magnet, the default location of the new Magnet is the location of the selected magnet.

When the Selection Set is empty or the entity in it is not a magnet, the default location of the Magnet is at u = 0.5, v = 0.5.

Relative to: If you want to make the Magnet relative to (offset from) another magnet, which must be on the same surface, check the box and specify the parent magnet. If you change the parent magnet, or if you change between absolute/relative coordinates, SurfaceWorks asks if you want to "Preserve Absolute Location" or "Preserve Offset Value" of the Magnet.

Example: Magnets.ms2

There are four magnets on this Tangent Boundary Surface: ‘basis’ (red) has absolute coordinates. It is the parent Magnet for the three relative Magnets ‘relmag1’, ‘relmag2’, and ‘relmag3’ (all three magenta). If the shape of ‘roof’ is changed (e.g. in the middle figure, by dragging the control point, pt6, outward), the absolute position of the magnets will change, but they will stay on the surface at the same u,v surface parameters. If ‘basis’ is moved (e.g. as in the bottom figure), the relative Magnets will move as well, always maintaining their relative u,v offsets to ‘basis’.



Magnets: initial model (top); with surface widened in middle (middle); with ‘basis’ moved leftward (bottom).

See also: Projected Magnet, Ring, Arc-length Ring, XYZ Ring

System Magnets

The symbols *00, *01, *10, and *11, which have been used in the past for "system graphs", can now be used also for "system magnets", as parents for certain snake entity types. *00 means the u=0, v=0 corner of a surface, *01 means the u=0, v=1 corner, etc. (This is parallel to the usage of *0 and *1 as "system beads", meaning the t=0 and t=1 ends of a curve.)

System magnets can be selected in the Entities Manager, by expanding the System heading. They are eligible as parents of the following snake types: Arc Snake, B-

Spline Snake, C-Spline Snake, Foil Snake, Geodesic Snake, Line Snake, and NURBS Snake.

Mirrored Curve

Parents and characteristic properties

Curve = curve entity. parent curve
Mirror = plane, Line or point entity

Description

A Mirrored Curve is a curve created by reflecting a given parent Curve across a specified Mirror entity. If Curve is later changed, the Mirrored Curve will be automatically changed to remain a mirror-image. Mirrored Curves are often useful in constructing models having partial or complete mirror symmetry with respect to some plane.

Mirror can be a plane, Line, or point entity:

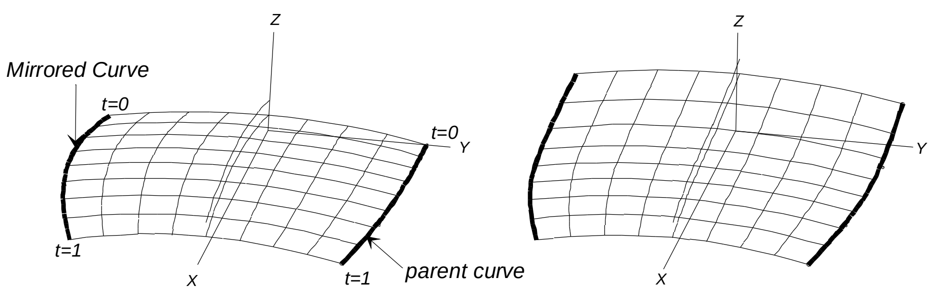
When Mirror is a plane, each projection line is perpendicular to the plane.
When Mirror is a line, each projection line is perpendicular to the line.
When Mirror is a point, each projection line passes through that point.

For comparative information about mirrored and projected entities, see "Understanding SurfaceWorks Entities - Collective Entity Information - Projected and Mirrored Entities".

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is distributed the same as on the basis curve.

Example: MirrorCurve.ms2

'side' is a B-spline Curve defined by four points. The plane on the centerline of our 'roof' is the default *Y = 0 centerplane. The curve 'center' is a Projected Curve in the centerplane. Mirrored Curve 'mirror' is created using parent Curve 'side' about Mirror plane *Y = 0 (left figure). The surface is a B-spline Lofted Surface using Control curves 'side', 'center', and 'mirror'.



MirrorCurve: initial model (left); model with parent curve modified (right).

In the right-hand figure, point ‘p1’ was moved to alter the curve ‘side’. Both the Projected Curve ‘center’ and the Mirrored Curve ‘mirror’ have changed shape accordingly.

See also: Projected Curve, Mirrored Point, Mirrored Surface, Projected Snake

Mirrored Point

Parents and characteristic properties

Point = point entity. parent point
Mirror = plane, Line or point entity

Description

A Mirrored Point is a point entity created by reflecting a given parent point across a specified Mirror entity. If Point is later moved, the Mirrored Point will be automatically relocated to remain in the mirror-image position. Mirrored Points are often useful in constructing models having partial or complete mirror symmetry with respect to some plane.

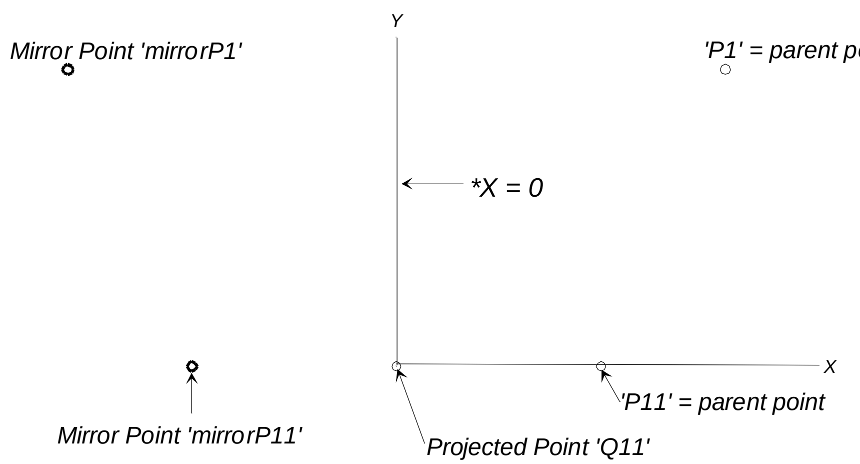
Mirror can be a plane, Line, or point entity:

When Mirror is a plane, each projection line is perpendicular to the plane.
When Mirror is a Line, each projection line is perpendicular to the Line.
When Mirror is a point, each projection line passes through that point.

You are not permitted to edit the location of a Mirrored Point, or drag it, as with most point entities, because it takes its position from that of the parent point.

For comparative information about mirrored and projected entities, see ”Understanding SurfaceWorks Entities - Collective Entity Information - Projected and Mirrored Entities”.

Example: MirrorPt.ms2



MirrorPt in Home view.

The first part of this example is the same as the first part of the ProjPoint example for Projected Point: ‘P1’ and ‘P11’ are Points, and ‘Q11’ is the projection of ‘P11’ onto *X = 0 (included for comparison of projected and mirrored points).

The two new points are ‘mirrorP1’ (green), the mirror image of ‘P1’ (green), and ‘mirrorP11’ (red), the mirror image of ‘P11’ (red).

See also: Point, Offset Point, Projected Point, Mirrored Curve, Mirrored Surface

Mirrored Surface

Parents and characteristic properties

Surface = parent surface
Mirror = plane, Line or point entity

Description

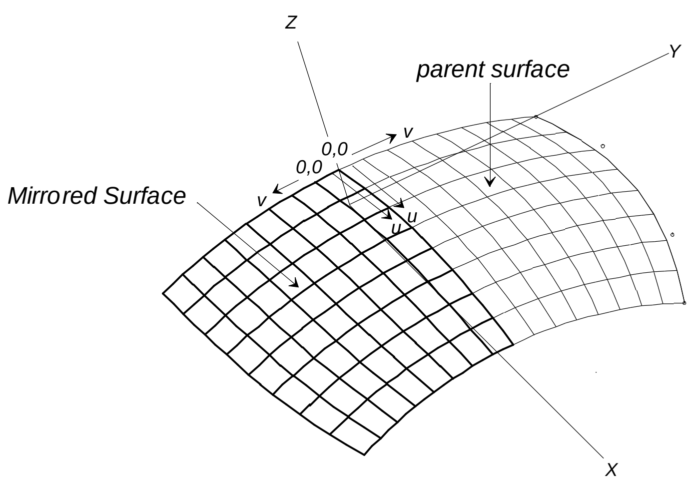
A Mirrored Surface is a surface entity created by reflecting a given parent Surface across a specified Mirror entity. If Surface is later changed, the Mirrored Surface will change to remain a mirror-image. Mirrored Surfaces are often useful in constructing models having mirror symmetry with respect to some plane.

Mirror can be a plane, Line, or point entity:

When Mirror is a plane, each projection line is perpendicular to the plane.
When Mirror is a Line, each projection line is perpendicular to the Line.
When Mirror is a point, each projection line passes through that point.

For comparative information about mirrored and projected entities, see “Understanding SurfaceWorks Entities - Collective Entity Information - Projected and Mirrored Entities”.

Example: MirrorSurf.ms2



MirrorSurf

‘roof1’ (green) is an Arc Lofted Surface of type “Arc: tangent at curve1 end”. ‘roof2’ (blue) is the Mirrored Surface of ‘roof1’ reflected across the *Y=0 plane.

See also: Mirrored Point, Mirrored Curve

NUB-spline Fitted Curve

Parents and characteristic properties

Degree = B-spline type

Knotlist = see below
No. of control pts = integer; number of control points to use for fit (0 if free)
Log-tolerance = log (base 10) of tolerance
Curve = basis curve

Description

Compared with a B-spline Fitted Curve, the only thing new here is the Knotlist. There are 3 alternatives for this entry:

1 '*' (the "default" knotlist) - This allows the fitting to use the best set of non-uniform knots it can come up with.

2 '*UNIFORM' - a new "predefined" knotlist, available in the System Objects: this forces the fitting to use uniform knots, essentially duplicating the function of B-spline Fitted Curve.

3 You can choose a KnotList or KnotList2 object containing a suitable specified list of knots.

As in the B-spline Fitted Curve, for No. of control pts, there are two basic choices:

A No. of control pts can be a positive integer (greater than *Degree*), specifying a definite number of control points to use for the fit.

B No. of control pts can be 0, which signifies "increase the number of control points until tolerance is met".

Since there are 3 alternatives for Knotlist and 2 alternatives for No. of control pts, altogether there are 6 combinations which produce different behaviors for the

NUB-spline Fitted Curve entity:

1A. '*' knotlist, combined with positive No. of control pts: The fitting will use exactly No. of control pts control points, and will use the best non-uniform knots it can generate.

1B. '*' knotlist, combined with No. of control pts = 0: The fitting will increase the number of control points (from type+1 up to the number of data points), until tolerance is met. At each step, it will optimize the non-uniform knotlist.

2A. '*UNIFORM' knotlist, combined with positive No. of control pts: The fitting will use exactly No. of control pts control points and uniform knots.

2B. '*UNIFORM' knotlist, combined with No. of control pts = 0: The fitting will increase the number of control points (from type+1 up to the number of data points), until tolerance is met. At each step, it will use uniform knots.

3A. KnotList object, combined with positive No. of control pts: The fitting will use exactly No. of control pts control points, along with the specified KnotList. In this case, there is a required relationship between No. of control pts, Degree, and number of knots: the number of interior knots (i.e., knots which are > 0 and < 1) = No. of control pts + Degree -1.

3B. KnotList object, combined with No. of control pts = 0: This case results in an error condition; because of the relationship noted under (3A); the number of control points is not free to escalate when the type and number of knots are both specified.

Tools>Measure>Clearance works with NUB-spline Fits (as with B-spline Fits); with a single NUB-spline Fitted object selected, it reports the number of control points used and the goodness-of-fit RMS and worst-point statistics.

Tools>Special>Freeze Fit works with NUB-spline Fits just as for B-spline Fits. They will generate a NURBCurve and its control points (or NURBSnake and its control magnets) as freestanding SurfaceWorks objects. When the fit is non-uniform, it generates a KnotList in addition.

See also: NUB-spline Fitted Snake, NUB-spline Fitted Surface

NUB-spline Fitted Snake

NUB-spline Fitted Snake

The NUB-spline Fitted Snake entity is very similar to the NUB-spline Fitted Curve. Its parent must be a snake, and the resulting approximation is in the u,v, space of the supporting surface.

Its control points are u,v locations on the supporting surface. The “tolerance” in logtolerance is a dimensionless value in u,v-space, rather than a distance.

See also: NUB-spline Fitted Curve, NUB-spline Fitted Surface

NUB-spline Fitted Surface

Parents and characteristic properties

u-Degree = B-spline degrees for u v direction: 1, 2, 3, etc.
u-Knotlist = see below
No. of control pts = integer; number of control points to use for fit (0 if free)
v-Degree = B-spline degrees for v direction: 1, 2, 3, etc.
v-Knotlist = see below
No. of control pts = integer; number of control points to use for fit (0 if free)
Log-tolerance = log (base 10) of tolerance
Surface = basis surface

Description

The “NUB-spline Fitted” entities are essentially replacements for the “B-spline Fitted” entities. B-spline Fitted Curve, B-spline Fitted Snake and B-spline Fitted Surface - with substantially greater flexibility. The “NU” stands for “non-uniform”. The B-spline Fitted's still exist for backwards compatibility, but the NUB-spline Fitted's do everything the B-spline Fitted's did, and much more.

The NUB-spline Fitted Surface entity works similarly to NUB-spline Fitted Curve, but it approximates a surface.

The fittings in the u- and v-directions are relatively independent. As with NUB-spline Fitted Curve, there are 3 alternatives for each Knotlist entry:

1 ‘*’ (the “default” knotlist) - This allows the fitting to use the best set of non-uniform knots it can come up with.

2 '*UNIFORM' - a new "predefined" knotlist, available in the System Entities; this forces the fitting to use uniform knots, essentially duplicating the function of B-spline Fitted Surface.

3 You can choose a KnotList or KnotList2 object containing a suitable specified list of knots

and two alternatives for each No. of control pts:

A No. of control pts can be a positive integer (greater than *type*), specifying a definite number of control points to use for the fit.

B No. of control pts can be 0, which signifies "increase the number of control points until tolerance is met".

These can be used in any combination, except **3B**. (No. of control pts = 0 can't be used in combination with a specified KnotList object, because of the fixed number of knots.)

One common reason to use a NUB-spline Fitted Surface prior to exporting IGES is to control the characteristics and quality of NURBS approximation on a surface-by-surface basis.

Another reason is to achieve exact matching along shared surface edges in the exported file. For example, suppose we have 'topside' and 'bottom' surfaces on a single-chine hull, joining each other exactly along the chine; suppose the u-direction is longitudinal and the v-direction is transverse on each surface. If we just export the 'topside' and 'bottom' surfaces through IGES with a global tolerance, we are very likely to come out with a different number of control points on the two exported surfaces. For example, the 'topside' might meet the tolerance with 12 x 5 control points, while 'bottom' gets 10 x 7. Two NURBS curves with different numbers of control points and different knots are identical only under extremely unlikely conditions, so there will almost certainly be gaps and overlaps between the two IGES surfaces that result.

However, if we first make two NUB-spline Fitted Surfaces 'nub_topside' and 'nub_bottom' using the following conditions:

- same ncu-specifier (say, 12 for this example)
- same knots for the u direction
- same number of u-divisions x u-subdivisions on the base surfaces

then we are guaranteed that the two fitted surfaces will join exactly along their common edge. Since there is no further approximation in the export of the NUB-spline Fitted Surfaces, the two exported surfaces will join exactly in the IGES file, and in the receiving application.

Tools and NUBFit Surf

Tools>Measure>Clearance works with NUB-spline Fitted Surface (as with B-spline Fitted Surface). With a single NUB-spline Fitted Surface selected, it reports the number of control points used and the goodness-of-fit RMS and worst-point statistics.

See also: NUB-spline Fitted Curve, NUB-spline Fitted Snake

NURBS Curve

Parents and characteristic properties

Degree = B-spline type: 1 = linear, 2 = quadratic, 3 = cubic, etc.
Knotlist = name of a KnotList object or '*' for uniform knot spacing
Control points = control points
Weights = corresponding weights

Description

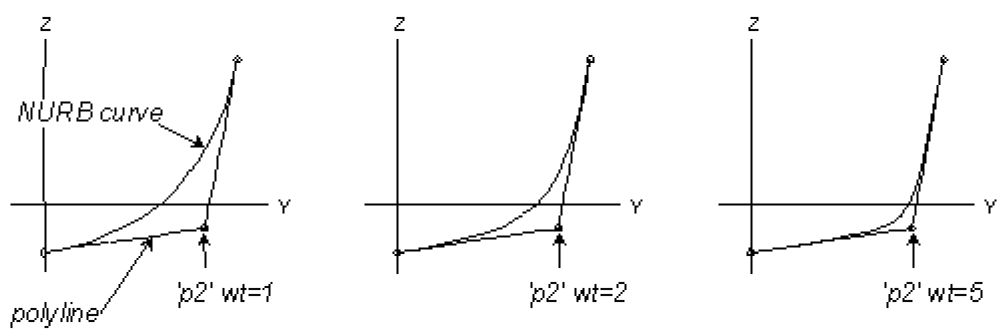
The NURBS Curve is a non-uniform rational B-spline curve. The Control points guide or shape the surface in much the same way as the control points shape a B-spline Curve. In general, the curve does not interpolate any of its control points except the first and last.

Weights are the weights associated with each control point.

Knotlist controls the knot spacing. If not default '*' (uniform spacing), the Knotlist should have (number of control points) - (type) + 1 components, and the list should begin with 0 and end with 1.

The program's default relabel produces the "natural" labeling, in which the parameter t is concentrated in highly curved areas formed either by closely spaced control points or by high weights.

Example: nurbcrv1.ms2



NURBCRV1.MS2 in <x> view: initial model (left); 'p2' weight edited (middle and right).

'midsection' is a NURBCurve formed by three points 'p1', 'p2', and 'p3', all of which have wt = 1 (left figure).

In the middle figure, the weight for 'p2' has been changed from 1 to 2, pulling the curve deeper. In the right-hand figure, the weight for 'p2' has been increased to 5, giving 'midsection' a tight bilge curve.

See also: B-spline Curve, NURBS Snake, NURBS Surf, B-spline Surface

NURBS Snake

Parents and characteristic properties

Degree = B-spline type: 1 = linear, 2 = quadratic, 3 = cubic, etc.
Knotlist = name of a KnotList object or '*' for uniform knot spacing
Control magnets = control magnets

Weights = corresponding weights

Description

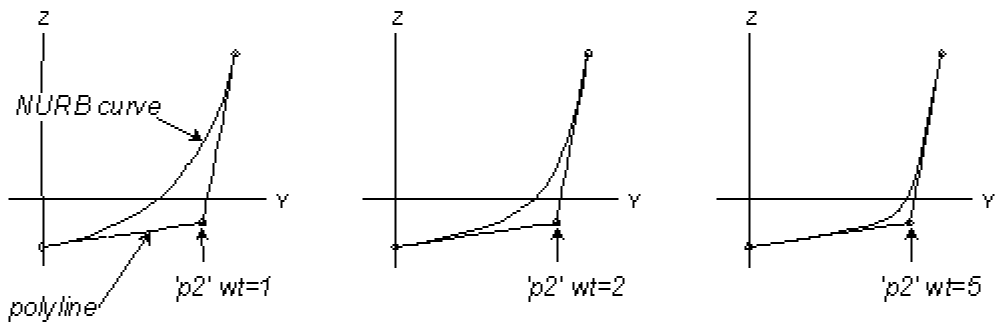
A NURBS Snake is a continuous curve lying in a surface and defined by a series of magnets for Control magnets. The magnets must all lie in the same surface. The curve ends at the two end Control magnets (first and last), but in general does not pass through the others (unless snake is type-1). Instead, it is “molded” or “shaped” by them. In the u-v parameter space, the NURBS Snake is a true NURBS Curve.

Weights are the weights associated with each Control point.

Knotlist controls the knot spacing. If not default ‘*’ (uniform spacing), the knotlist should have (number of control points) - (type) + 1 components, and the list should begin with 0 and end with 1.

The program’s default relabel ‘*’ produces the “natural” labeling, in which the parameter t is concentrated in highly curved areas formed either by closely spaced control points or by high weights, in the u-v parameter space.

Example: nurbsnk1.ms2



NURBSNK1.MS2 in Lat 0, Lon 60 orthographic view: initial model (top); with two weights edited (bottom).

‘cabin_side’ is the side of a boat cabin. ‘portlight’ is a type-2 NURBSnake formed by the seven magnets ‘csm1’ to ‘csm7’ which all lie on ‘cabin_side’ and all have wt = 1 (top figure). If the shape of ‘cabin_side’ is modified, ‘portlight’ will change its position in space but will remain always on ‘cabin_side’, in a corresponding position.

The lower figure shows the effect of increasing the weights of ‘csm2’ and ‘csm7’ to 3.

See also: B-spline Curve, NURBS Curve, NURBS Surface, B-spline Surface

NURBS Surface

Parents and characteristic properties

u-Degree, vDegree = B-spline types for u and v directions: 1, 2, 3, etc.
u-Knotlist and v-Knotlist = names of two KnotList objects or ‘*’ for uniform knot spacing
No. of u control pts = number of control points in u-direction
Control points = total number of control points
Weights = corresponding weights

Description

The NURBS Surface is a rational B-spline surface. The control points form a quadrilateral **net** which guides or shapes the surface in much the same way as the control points shape a B-spline Curve (Fig. 1). In general, the surface does not interpolate any of its control points except the four at the corners of the net.

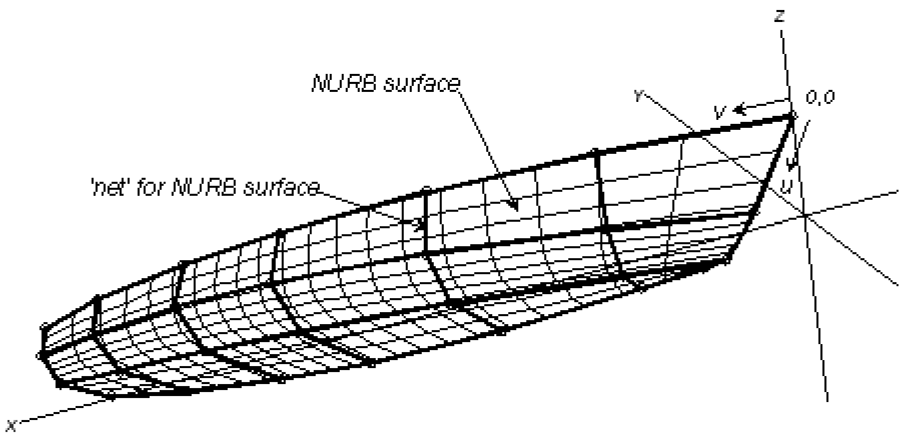


Fig. 1
A NURBS Surface and its net.

The edges are all rational B-spline curves using control points from the four edges of the net:

v = 0 uses the first ncu control points

v = 1 uses the last ncu control points

u-Knotlist and v-Knotlist control knot spacing. If not default '*' (uniform spacing) for each list, the knotlists should have (ncu - utype + 1) and (ncv - vtype + 1) components respectively, and each list should begin with 0 and end with 1.

Control points is the total number of control points for the surface and must be the product of two integers. One of these integers is specified as the No. of u control pts which is the number of control points in the u-direction. The other integer is the number of control points in the v -direction.

Weights are the weights associated with each Control point. If Weights are all 1 you get the plain vanilla B-spline Surface. Increasing weight on one control point draws the surface (and the parameter lines) toward that control point. In Figure 2, point 'P32' has been given a weight of 3, while all others remain at 1. This pulls the surface (and the parameter lines) toward point 'P32'. Below the profile view, the body plan view compares the X = 10 contour of the original, evenly-weighted model to that of the model with 'P32' weight = 3.

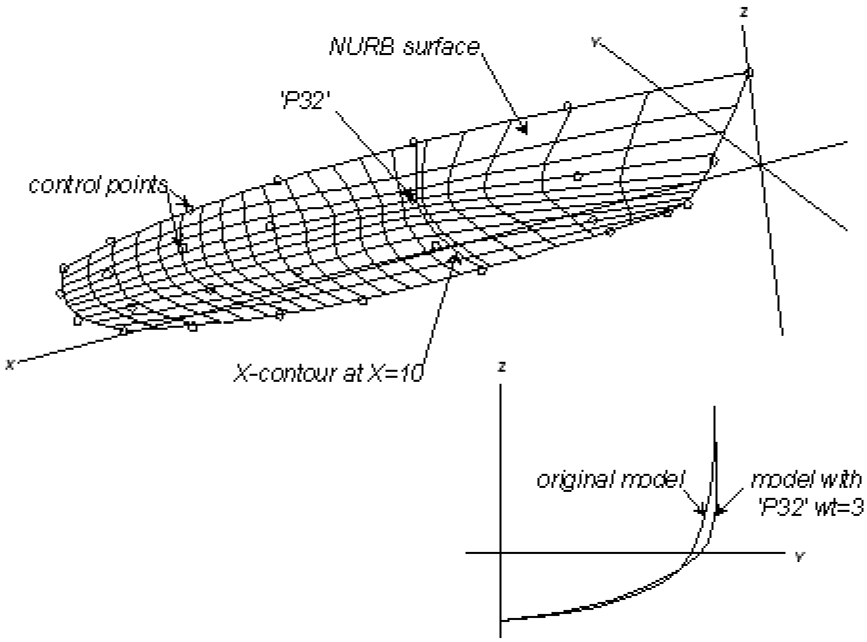


Fig. 2
A NURBSurf with the weight on one point ('P32') increased to 3.

Examples: nurbs7a.ms2, nurbs7c.ms2.

'hull' is a NURBS Surface formed by 28 points ('P11' to 'P47') with all weights equal to 1. This is the same hull as in Figure 1 above, but to make that figure, we used 'Show net' = 'True' plus u-constant and v-constant lines. The u-direction B-splines are type-2 (quadratic) and the v-direction B-splines are type-3 (cubic). The two default knotlists distribute the knots uniformly in both the u and v directions. There are 4 control points in the u-direction (and 7 in the v-direction).

Now let's see what happens when we change knot spacing in the v-direction. This means we need to add the definition of a knotlist to the model. Since in the longitudinal direction on this hull surface we use cubic splines (vtype = 3) with ncv = 7 columns of control points, then the knotlist for the v-direction (which we named 'knotsv') needs 7 - 3 + 1 = 5 entries, for example { 0 .25 .5 .75 1 }.

This particular knotlist would actually produce uniform B-splines; i.e., it would have the same effect as using the default '*' for knotlist2 in the NURBS Surface description (Fig. 3, top).

But using:

{ 0. .45 .50 .55 1.00 } would allow a region of high longitudinal curvature near the middle (v = .5) of the hull

{ 0. .50 .50 .50 1.00 } would actually permit a slope discontinuity (in this case, a crease or bend) to run around the hull at v = .5, as in the model nurbs7c.ms2 (Fig. 3, bottom). Note that at most type + 1 sequential knots can have identical values.

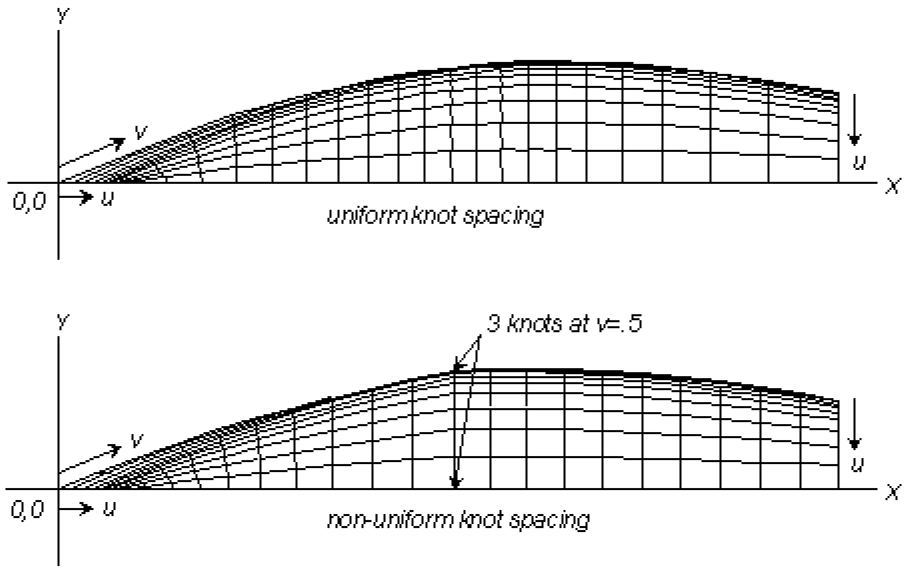


Fig. 3. Uniform (top; NURBS7A.MS2) and non-uniform (bottom; NURBS7C.MS2) knot spacing on a NURBSurf; view here is Lat 90, Lon -90 orthographic.

See also: B-spline Curve, B-spline Surface, B-spline Lofted Surface, C-spline Lofted Surface

Offset Curve

Parents and characteristic properties

Snake = basis snake
Offset1 and Offset2 = offset distances at ends of snake
Graph = name of a B-spline Graph object or '*' for default graph

Description

An Offset Curve is constructed from a snake by taking each point on the snake and moving it an offset distance along the normal to the surface at that point. Offset1 and Offset2 are signed decimal numbers which specify the offset distance at the t=0 and t=1 ends of Snake, respectively.

The Graph supplies a "blending function" which controls how the end offset values Offset1, Offset2 are distributed along snake to find points on the OffsetCurv. A point on the OffsetCurv at parameter value t is located by these steps:

1. t is passed through the specified relabel function to get the natural parameter t'.
2. Graph is evaluated at parameter t', resulting in a value B of the blending function.
3. Snake is evaluated at t'.
4. The resulting point is displaced along the normal to the surface by a weighted combination of the end offsets: (1-B)Offset1 + B Offset2.

The default Graph '*' in this context is a straight line from 0,0 to 1,1 (just like the default relabel '*'); this produces a linear blending of the end offsets.

In many cases (including the default) the Graph will run from 0,0 to 1,1, i.e. the first and last Graph values will be 0. and 1. Whenever this is the case, the offset distance starts at Offset1 and ends at Offset2. However, this is not required.

Should you want to directly specify the distribution of offset distance by the shape of the graph, here's a convenient way:

1. Set Offset1 to 0.
2. Set Offset2 to the final offset value.
3. Shape the Graph so it expresses the desired distribution of offset distance and ends with 1.

For a constant offset, specify the same number for Offset1 and Offset2, and use the default Graph.

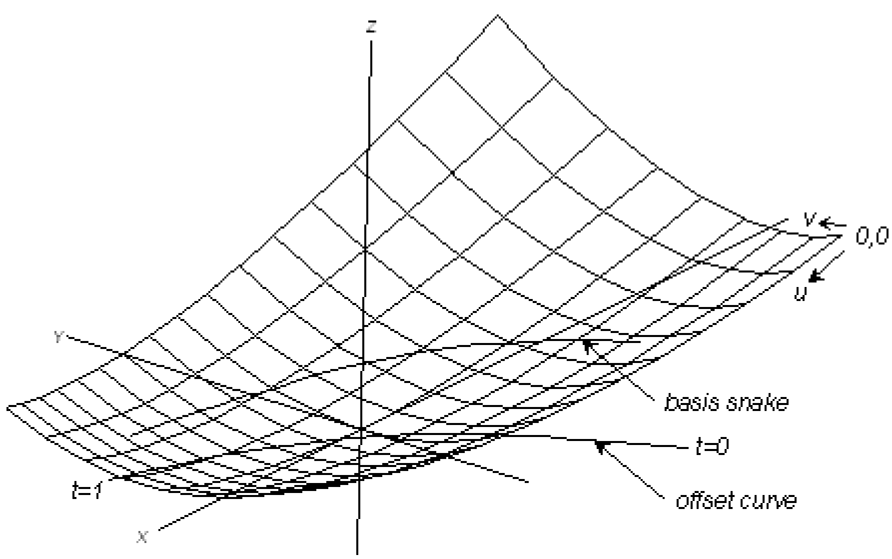
The natural labeling of an Offset Curve is identical to that of its basis Snake.

Note: If you change Orientation for the surface, the OffsetCurv will move to the other side of the surface.

Example 1 Offset Curve.ms2

Example 1: offsetcv.ms2

The normal orientation switch (Orientation) for the surface 'patch' is 0 (zero), which means the positive normal points up as you look at the model from this Lat -30, Lon 40 viewpoint. 'basis_snake' (magenta) lies in 'patch, and 'offset_curve' (blue) is offset in the negative normal direction from the snake, 1 unit from its t=0 end and .5 unit from its t=1 end.

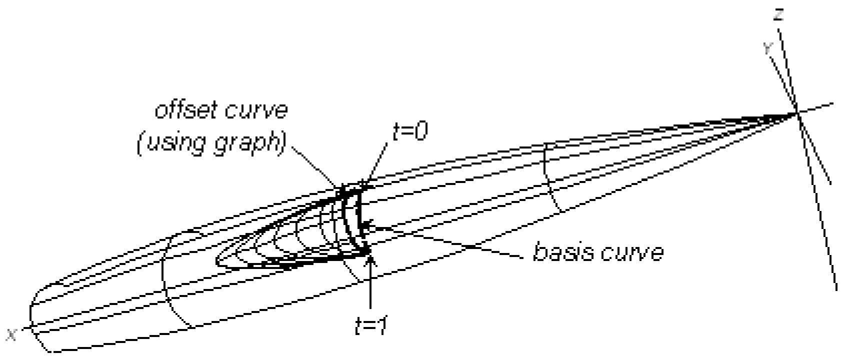


OFFSET CURVE.MS2 at Lat -30, Lon 40.

Example 2 Inlet.ms2

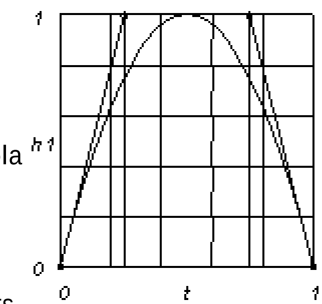
Example 2: inlet.ms2

In this example, an OffsetCurv with a graph is used to make the opening of an air inlet for the JET model.



INLET.MS2 at Lat -30, Lon 60.

'n1' (magenta) is a LineSnake on 'fuselage'. OffsetCurv 'lip' (cyan) is made from 'n1' with 0 (zero) offset specified at t = 0 and 1.00 ft offset at t = 1. BGraph 'h1' is type-2 with 4 values: 0, 1, 1, 0. This forms a parabola (see graph at right) which specifies the distribution of offset: zero at both ends; maximum of 1.0 in the center.



The inlet is a Blister Surface using 'lip' for its apex (= curve).

See also: OffsetPt, OffsetSurf

Offset Point

Parents and characteristic properties

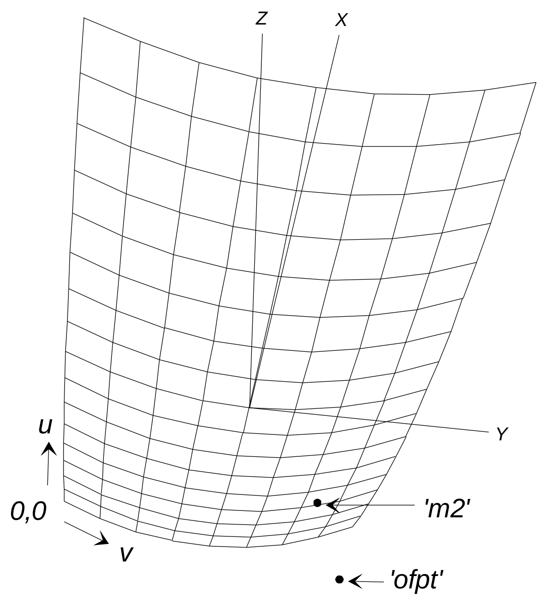
Magnet = magnet or ring. basis point
Offset = the offset distance from magnet

Description

An Offset Point is a point entity located a specified distance offset along the normal to a surface, at a location designated by a Magnet (or ring). Offset is any decimal number. If Offset is positive, the Offset Point lies along the positive normal to the surface. Clicking the **Reverse Offset** button multiplies the number in Offset by -1 and displays this new result.

Example: OffsetPt.ms2

The positive normal for the surface 'patch' points away from you as you look at the model from this viewpoint. The Offset Point 'ofpt' (blue) is located at a distance of 1.000 unit in the negative normal direction from the magnet 'm2' (green).



See also: Point, Offset Surface

Offset Surface

Parents and characteristic properties

Surface = parent surface
Offsets = the offset distance to be used

Description

An Offset Surface is formed by offsetting each point of the parent surface along the direction normal to the basis surface. To create an Offset Surface, you specify the parent Surface and the magnitude of the Offset.

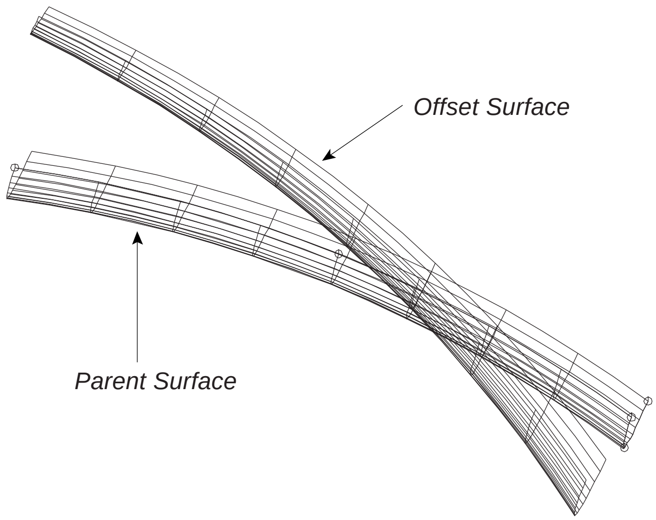
If you want to have different offsets for each of the corners (u=0, u=1, v=0, v=1) then right click on the Offsets field and append or Insert 3 rows and input your desired values.

Use Orientation to move the Offset Surface to the other side of the parent surface.

For most accurate results, give the Offset Surface the same divisions as the basis surface.

Note: You may get unexpected results if the offset is large and / or if the parent surface is twisted.

Example: OffsetSurf



OffsetSurf

‘patch’ (dark green) is the parent surface for the Offset Surface ‘offset’ (blue). The v=0 edge of 'offset' is offset from patch by -1. The v=1 edge is offset from 'patch' by 2. If you change Orientation to Reverse these values will change sign.

See also: Offset Point

Offset TriMesh

Parents and characteristic properties

Surface/TriMesh = parent surface or TriMesh
Offsets = the offset distance to be used

Characteristic data

An Offset TriMesh is a triangle mesh created by offsetting each node of the parent TriMesh by a uniform distance in the local normal direction. It is analogous to an Offset Surface (with constant offset).

If Offset is positive, the offset is in the direction of the parent surface/trimesh's positive normal. If negative, the offset is in the opposite direction.

Example: OfstTriMesh2.ms2

The parent TriMesh I0 (green) has a breakline across it. The Offset is 0.10 (positive), so the offset is in the direction of I0's positive normal. The OfstTriMesh I1 (cyan) preserves the sharp breakline.

Plane

Characteristic data

Offset Plane Data = Data required for Offset Plane. 'Plane', 'Offset'.
Point Plane Data = Data required for 2 and 3 Point Planes. point entity. '1st Point', '2nd Point', '3rd Point'.

Description

There are 3 types of planes available to users: Offset, 2-Point and 3-Point. Planes are used in the construction of some entities, in particular projected and mirrored entities, but planes are not parametric surfaces — you cannot locate magnets and snakes on them.

Note: The plane of the XY axii of any **Frame Entity** can always be used as a plane

Offset Plane.ms2

The Offset Plane is an infinite plane at constant offset value from its parent plane. The positive normal is in the same direction as its parent's positive normal. A visible Offset Plane is displayed as a rectangle, sized slightly larger than the rectangular box enclosing the current wireframe model. To create this plane, pick the plane that you want to be parallel to and then the offset distance from this plane. The Reverse Offset check box changes the side of the plane on which the offset plane lies.

Note: **Offset Plane made while using** SurfaceWorks **under SolidWorks** (not stand alone) **that has *Z=0 as its parent** will be offset in the opposite direction than expected. For example, if you create an offset plane with *Z=0 as its parent and give it an offset of 1.0, it will actually intersect the Z axis at -1.0

2-Point Plane.ms2

The 2-Point Plane is an infinite plane. A visible 2-Point Plane is displayed as a rectangle, sized slightly larger than the rectangular box enclosing the current wireframe model.

The plane passes through the 1st point and is oriented such that the 2nd point lies along the positive normal through the 1st point. Using a relative Point for the 2nd point (with the 1st point as its basis point) is an easy way to directly control the normal direction for the 2-Point Plane entity. The distance of the 2nd point from the 1st point has no effect on the plane (however, it has to be greater than zero).

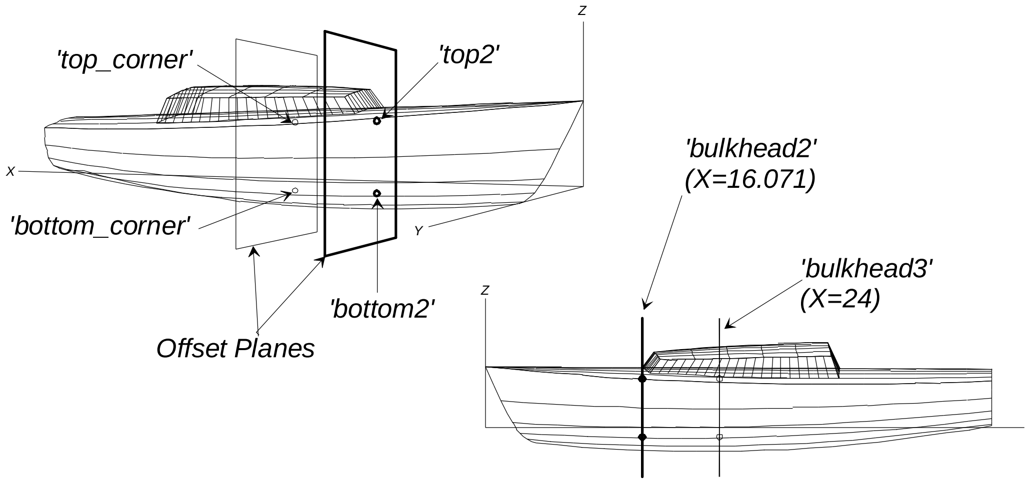
3-Point Plane.ms2

The 3-Point Plane is an infinite plane. A visible 3-Point Plane is displayed as a rectangle, sized slightly larger than the rectangular box enclosing the current wireframe model.

The plane passes through all three points. It is oriented according to a right-hand rule: if you are located somewhere along the positive normal, looking toward the plane, the triangle made by connecting the 1st point - 2nd point - 3rd point - 1st point is traversed in the counterclockwise direction.

Offset Plane Example: xplane.ms2

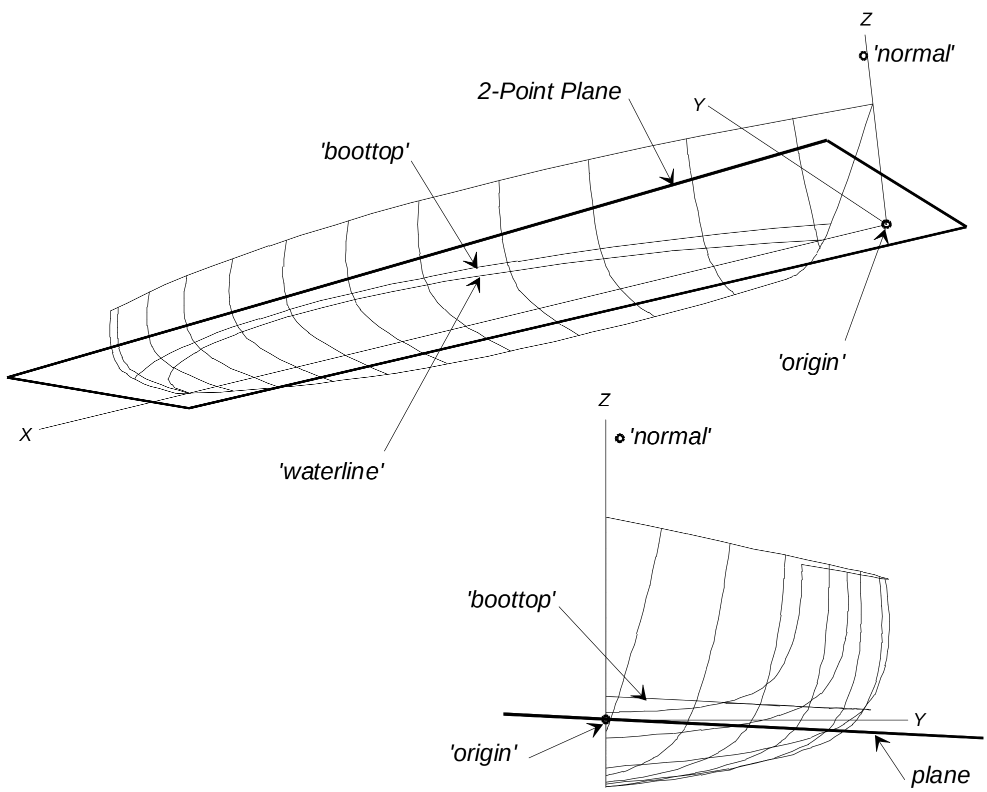
‘bulkhead3’ is an Offset Plane (yellow) parallel to the *X=0 plane located at X = 24. ‘top_corner’ and ‘bottom_corner’ (both yellow) are two corners of a door located in the plane of ‘bulkhead3’. ‘bulkhead2’ is another Offset Plane (magenta) parallel to the *X=0 plane located at X = 16.071. ‘top2’ and ‘bottom2’ (both magenta) are forward projections of ‘top_corner’ and ‘bottom_corner’ onto ‘bulkhead2’, locating a second door in this bulkhead.



xplane.

2-Point Plane Example: 2plane1.ms2

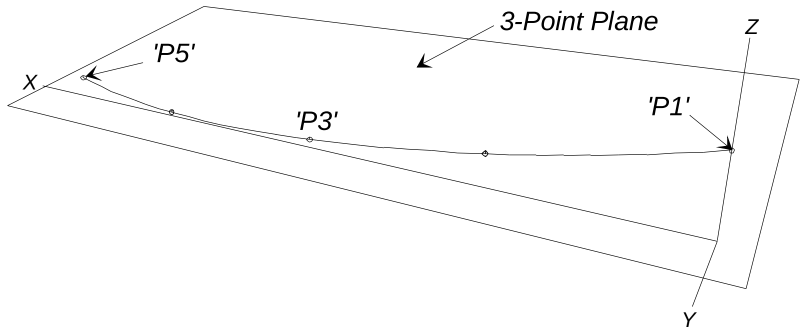
'hull' is a C-spline Lofted Surface formed by three master curves. 'plane' (white) is a 2-Point Plane defined by the 2 points 'origin' (1st Point) and 'normal' (2nd Point). The plane passes through 'origin' and its positive normal passes through 'normal'. 'plane' includes the X-axis and is inclined at about 3 degrees from horizontal. 'boottop' is the curve of intersection between 'hull' and a plane parallel to 'plane' and 0.4 units above it.



2plane1.

3-Point Plane Example: 3plane0.ms2

In this example, the 3-point plane ‘sheer_plane’ is used to make a planar sheerline. ‘P1’ - ‘P5’ are the roughed out control points for the sheerline. ‘P1’, ‘P3’, and ‘P5’ are used to define the Plane3. ‘Q2’ and ‘Q4’ are projected off ‘P2’ and ‘P4’ onto ‘sheer_plane’. ‘sheer’ is a C-spline Curve running through ‘P1’, ‘Q2’, ‘P3’, ‘Q4’, and ‘P5’ — it is a planar C-spline Curve because all its control points lie in the same plane.



3plane0.

Point

Parents and characteristic properties

Offsets = distance from Point in x, y, z directions of the Frame If Polar coordinates is checked, the values become lat, lon and rad of the Frame.
Parents, Point = point entity. the parent point from which to measure dx, dy and dz
Parents, Frame = the parent frame which is used to find the direction of dx, dy and dz

Description

A Point is a point entity in 3D space located by specifying its dx, dy and dz coordinates. Because every Point in SurfaceWorks has two parents: a Point and a Frame, the location of these coordinates depends on the Frame they are being measured in and the Point they are being measured from.

The default Point is the origin of whatever the Frame is. The default Frame is the global coordinate system.

Example: Points.ms2

Points shows the possible combinations of default and non-default parents for creating points. 'pt1' uses both defaults, so its position is measured from the origin of the global coordinate system. 'pt2' and 'pt3' both use the default Frame but have different Point parents. 'pt4's parents are the origin (default Point) of the Frame 'frame1', and 'frame1' itself. 'pt5' uses 'pt4' for its Point parent and 'frame1' for its measuring frame.

Points 'pt6' and 'pt7' are points with polar coordinates using the default Point and Frame. Points 'pt8' and 'pt9' have the same "lat", "lon" and "rad" as 'pt6' and 'pt7' but use 'pt1' and 'frame1' as their Point and Frame parents.

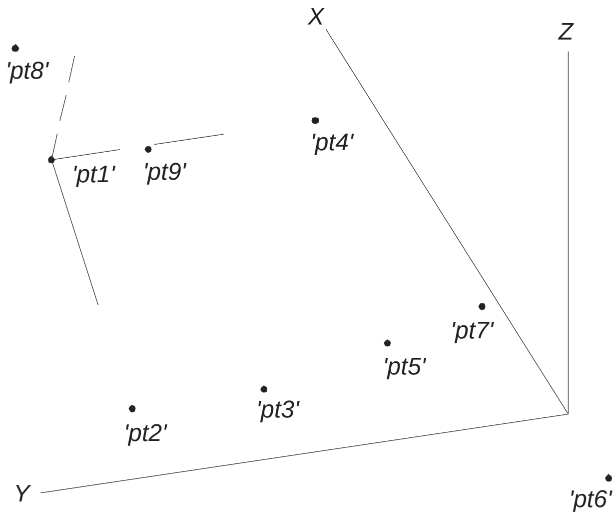
When you enter polar coordinates, SurfaceWorks converts them to Cartesian coordinates for display using the following formulae:

x = x0 - rad*cos(lat)*cos(lon)

y = y0 + rad*cos(lat)*sin(lon)

z = z0 + rad*sin(lat)

Where x0, y0 and z0 are the location of the Point parent and rad, lat and lon are the values specified. The angles are calculated in the frame used as the Frame parent.



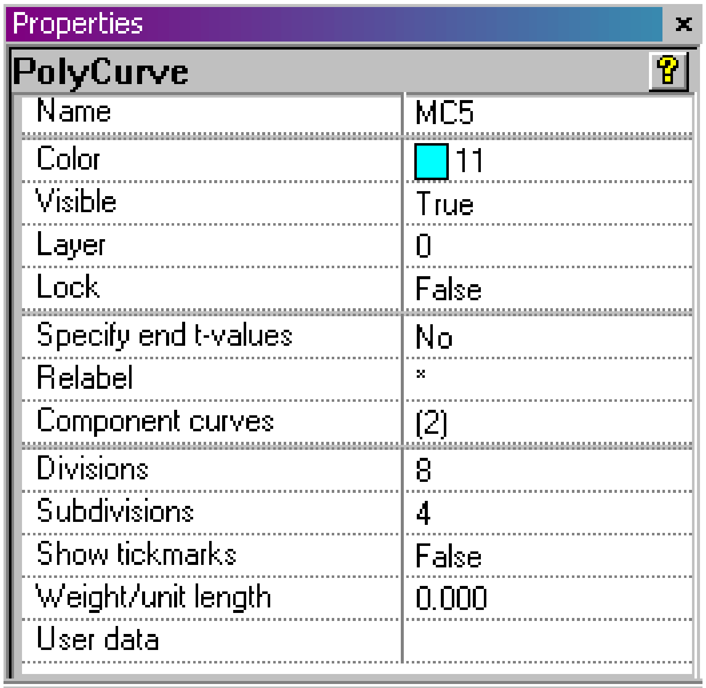
Points.

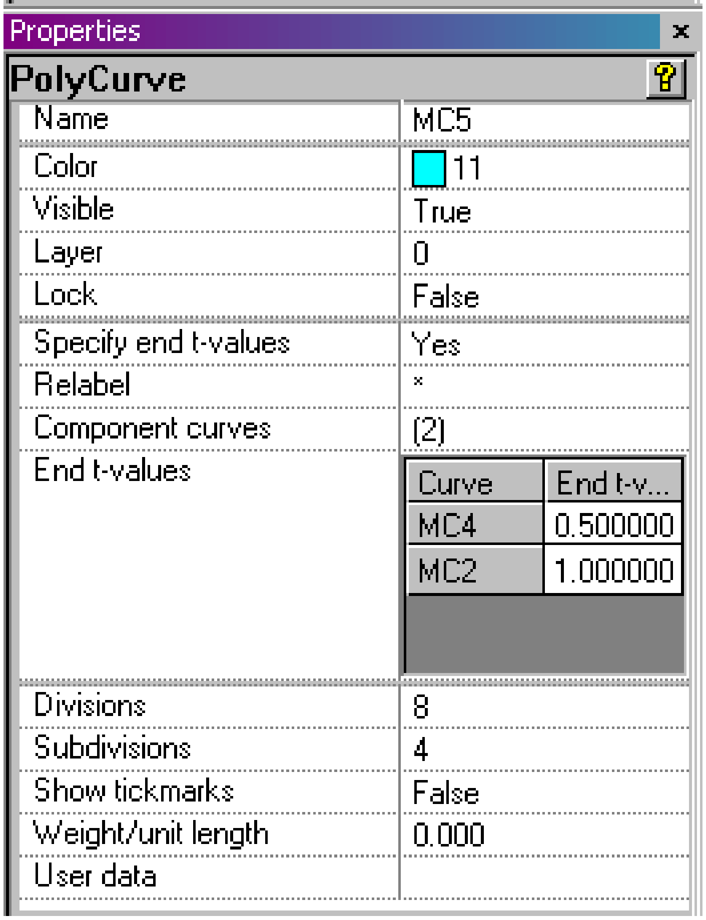
See also: Offset Point, Mirrored Point, Projected Point, Bead, Arc-length Bead, XYZ Bead, Magnet, Projected Magnet, Ring, Arc-length Ring, XYZ Ring

PolyCurve

Parents and characteristic properties

Specify end t-values = Yes / No
Component curves = curve entity. curve1, curve2 . . . curveN





Description

A PolyCurve is a single curve made by joining two or more parent Component curves end-to-end. You select the parent Component curves in the order in which they are to be assembled. They can be any point, curve, or snake entities.

The end of curve1 must be the same point as the start of curve2, etc. If this in not the case, an error will result.

An alternate version of PolyCurve may be required by some users for certain applications. The biggest difference is the ability to set the location of the ending t-values of the various segments of the PolyCurve. In this case continuous velocity is not assured through each knot, but this could be advantageous in some applications.

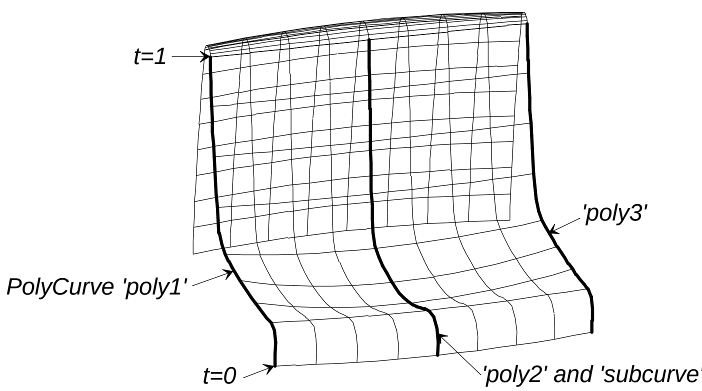
Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, which reflects the labeling of the component curves, but with continuous parametric velocity through each knot.

Example: Polycurve-Subcurve2.ms2

This model uses PolyCurves to make single curves from multiple curves and lines. This is needed because the parent curves for the lofted surface of the seat must each be a single curve.

The Component curves for the 'poly1' PolyCurve (cyan) are the parents Line 'xline6' plus Arcs 'x15', 'x14', 'x13', and 'x12' (all magenta), selected in that order. The first curve selected, 'xline6', is at the t=0 end of the PolyCurve.

The surface is an Arc Lofted Surface using three parent lofting curves: 'poly1' and 'poly3' (PolyCurves; both cyan), and 'subcurve' (blue), a SubCurve made from another PolyCurve 'poly2'. (Why the SubCurve? You can control the parameterization of a SubCurve made with three or more control beads — thus we were able to adjust the shape of the center of the seat to what we wanted. For details, see "Example 2: Polycurve-Subcurve2".



Polycurve-Subcurve2

See also: SubCurve

PolyGraph

Parents and characteristic properties

Component graphs = The graphs to join together
End t-values = t-values of the parent parameter space where the graphs meet

Description

A PolyGraph is two or more graphs strung end-to-end and reparameterized to the range 0-1.

This entity greatly increases the flexibility with which we can design graphs.

End t-values specifies where in the parameter space of the parent entity you want the graphs to join.

Examples

(1) Tapered mast. A typical mast taper is linear, starting at some point from 50 to 60% of the exposed length of the mast. Using B-spline Graphs, we could use Degree-1 and easily start a taper at 50% { 1. 1. .6 } or 60% { 1. 1. 1. 1. .8 .6 } but there's no way to get 58%, except making a 51-value graph with 20 carefully calculated values.

PolyGraph solution: Make 2 Degree-1 B-spline Graphs {1.0} and {1.0 0.6 } then make a PolyGraph with End t-values at .58.

(2) Tapered rubrail. A rubrail is likely to have constant section for 96% of its length, and all its taper in the last 2% at each end. This requires a B-spline Graph with 100 values.

PolyGraph solution: Make a Degree-2 B-spline Graph v0 { 0 1 1 } , a Degree-1 v1 { 1.}, and another Degree-2 v2 { 1 1 0} Then make a PolyGraph using v0, v1 and v2 and End t-values of .02, and .98.

(3) Complicated sweeps. Being able to mix segments of Degree-1 and Degree-2 B-spline Graphs into a single graph gives a lot of flexibility for molding ribs, ridges and other features often seen in plastic molded parts.

PolySnake

Parents and characteristic properties

Specify end t-values = Yes / No
snake1, snake2 .... snakeN = component snakes (t-orientation doesn't matter)

Description

A PolySnake is a single curve made by joining two or more other snakes (called the "component snakes") end-to-end. The component snakes must all lie in the same surface and be selected in the order in which they are to be assembled.

It is anticipated that the end of snake1 is actually the same point as one of the ends of snake2, etc. However, this is not enforced or checked. If you specify snakes that do not share endpoints, the moving point will jump discontinuously from the end of snake1 to the start of snake2 as the parameter t passes t_1, etc. (see next paragraph).

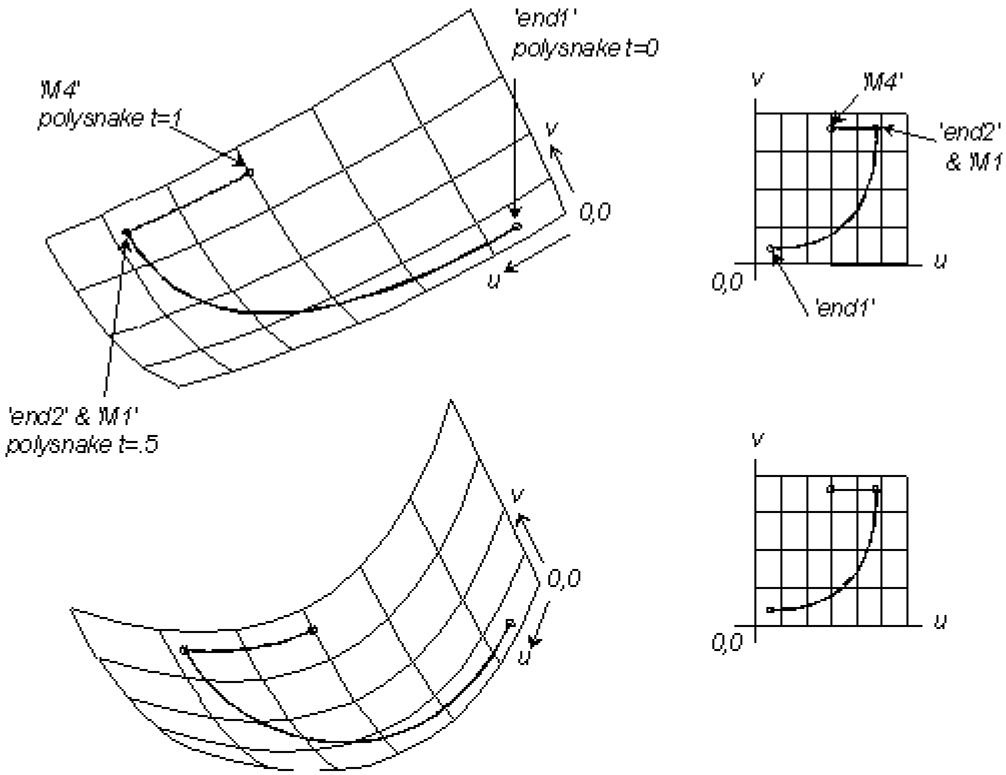
An alternate version of PolySnake may be required by some users for certain applications. The biggest difference is the ability to set the location of the ending t-values of the various segments of the PolySnake. In this case continuous velocity is not assured through each knot, but this could be advantageous in some applications.

relabel is used to relabel the snake. The program's default relabel '*' produces the "natural" labeling, in which the parameter t is t_1 at the end of the first component snake, t_2 at the end of the second component snake, etc., in the u-v parameter space.

Example: polysnk1.ms2

't_reversed' (cyan) is the same subsnake we created in the second example for the "SubSnake" entity description. 't_reversed' begins (t = 0) at 'end1' and ends (t = 1) at 'end2'. 'lsnake' (yellow) is a line snake that begins at magnet 'M1' and ends at magnet 'M4'. Both 't_reversed' and 'lsnake' lie in the surface 'patch'. 'end2' (t = 1 for 't_reversed') and 'M1' (t = 0 for 'lsnake') are at the same position. 'polysnake' (magenta) is formed by joining 't_reversed' and 'lsnake'. t = 0 for 'polysnake' is at 'end1', t = .5 is at the end of 't_reversed' ('end2'), and t = 1 is at the end of 'lsnake' ('M4').

In the lower figure, the shape of 'patch' has been changed, but you can see that 'polysnake' still lies in the surface.



POLYSNK1.MS2 at Lat -60, Lon 60: initial model (top); with 'patch' surface edited (bottom).

See also: PolyCurve, PolySurface, SubSnake

PolySurface

Parents and characteristic properties

surface1 ... surfaceN = component surfaces
u1, u2 ... 1 = transition values for the u parameter

Description

A PolySurf splices two or more surface objects together end-to-end to create a single surface. The principal reason for wanting to do this is to reduce the total number of surface meshes required to represent your model. For example, in some forms of computational fluid dynamics (CFD) analysis it is desirable to minimize the number of surface patches, to avoid complications of matching meshes and establishing neighbor connections between adjacent patches.

surface1 ... surfaceN are the surfaces to be joined, in order. To make a continuous PolySurf, the component surfaces must join on opposing edges, effectively in a single strip (Fig. 1). The orientation of u and v directions in each component surface is not significant. The u-direction on the PolySurf will always be along the strip, and the v-direction across the strip. The corner u=0, v=0 will always be at a corner of surface1.

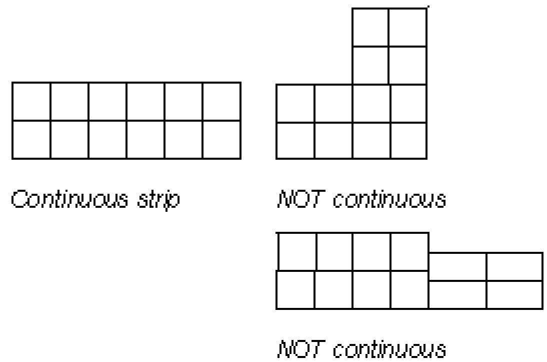


Fig. 1

u1, u2, ... are transition values for the u parameter, where one component surface ends and another begins. In general it is desirable to coordinate the transition u parameter values with the surface divisions. If your surface has nu total subdivisions (u-divisions x u-subdivisions) in the u-direction, each transition parameter value should be an integer multiple of 1/nu. This will assure that the surface tabulation will include a parametric line u=constant exactly along the junction of the component surfaces.

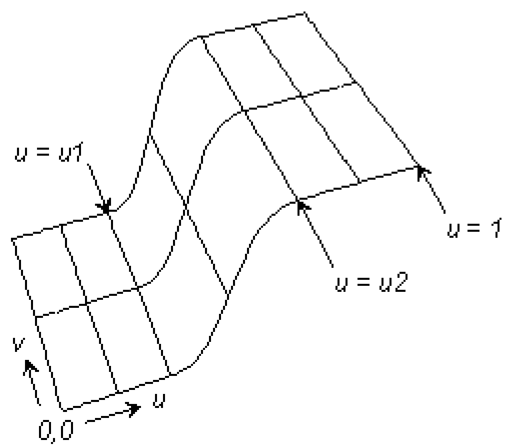


Fig. 2

Note: When you change the supports list for a PolySurf, its u-transition values are reset to even distribution — e.g. for a three-surface PolySurf, the u-transition values would be reset to .333, .667, and 1.0.

Example: wingpoly.ms2

‘polysurf’ (green) has 3 component surfaces: the wing-body fillet ‘wingroot’ (yellow), the wing subsurface ‘wing_sub’ (magenta), and the wingtip fillet ‘thetip’ (blue). ‘polysurf’ has a total of 12 divisions in the u-direction (u-divisions = 4 and u-subdivisions u-subdivisions = 3), and each transition parameter value is an integer multiple of 1/12 = .083333 (u1 = .083333 x 4; u2 = .083333 x 8).

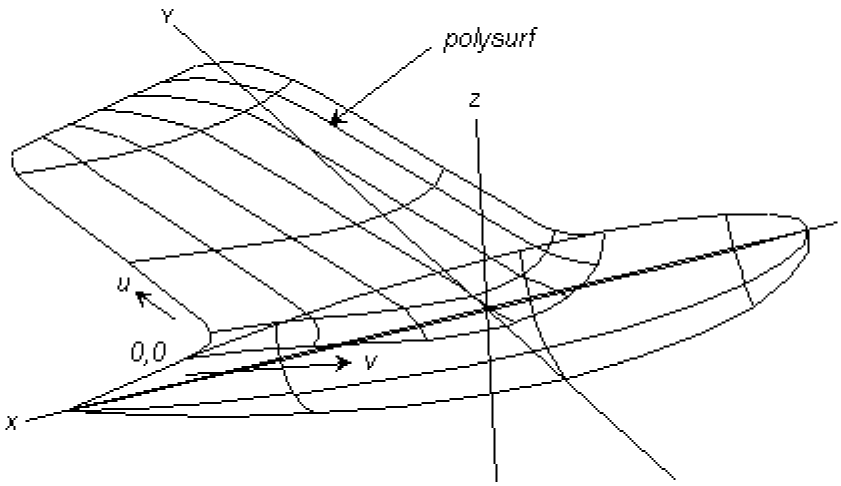


Fig. 3. WINGPOLY.MS2 at Lat -30, Lon 60 (‘thetip’, ‘wing_sub’, and ‘wingroot’ hidden).

See also

[PolyCurve]
[PolySnake]

See also: PolyCurve, PolySnake

PolyTriMesh

Parents and characteristic properties

degree = sub-division amount
Log-tolerance = Base-10 logarithm of distance tolerance
Triangle meshes = Parent list

Description

The Poly TriMesh entity type is two or more TriMesh objects assembled into a single TriMesh. The component TriMeshes currently need to have compatible triangulations, with nodes agreeing within a tolerance specified by the log-tolerance parameter.

Example PolyTriMesh1.ms2

Poly TriMesh ‘I0’ is assembled from two parent TriMeshes: 'tm1' (6 nodes, 4 triangles) and 'tm2' (9 nodes, 8 triangles). Since the two parents share 3 point entities as their common nodes, the value of Log-tolerance is not significant in this example.

Procedural Curve

Parents and characteristic properties

point = the moving point whose path is the ProcCurve
bead/ring/graph1 ... bead/ring/graphN = support(s) for point; also define the ProcCurve modeling process

Description

A Procedural Curve is defined by creating one point on the ProcCurve, then having SurfaceWorks repeat the point creation procedure for all positions of the supporting bead/ring(s) along their host curve(s) or snake(s).

point is any kind of point object constructed directly or indirectly from the 1 or more bead/rings.

The bead/rings have to be AbsBeads or AbsRings.

To generate the ProcCurve, the program varies the parameters of bead/ring1 from t=0 to t=1, varying the other bead/rings in unison, and the ProcCurve is the resulting path of point. If the bead/rings don't all have the same t, the program preserves the t offsets between them. Any graphs included as supports are evaluated at each parameter value t and substituted, in order, for any decimal data values in point.

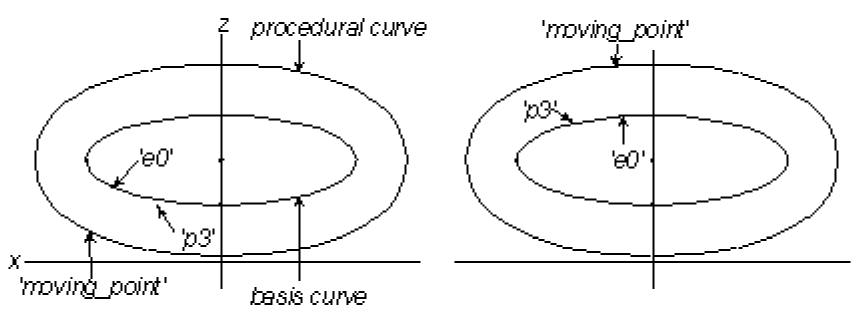
Example 1 Procedural Curve.MS2

'rim' (white) is the ProcCurve in this example. It is made by repeating the procedure of making 'moving_point' (yellow) for every position of 'e0' (green) along the basis curve 'c0' (cyan; an ellipse). In this case, moving point ('moving_point') is only indirectly based on bead ('e0'). Here are the details:

'e0' (green) is an AbsBead that lies on the type-1 Conic 'c0' (an ellipse). 'p3' (red) is a TanPoint offset from 'e0' by 1.5 units. 'moving_point' (yellow) is a RotatPoint based on 'p3' and rotated 90 degrees around the Line 'axis' (use the <x> view to see 'axis').

The program generates the ProcCurve by varying the parameter t for 'e0' from 0. to 1. — the resulting path of 'moving_point' is the ProcCurve, a curve offset exactly 1.5 units from 'e0' in the plane perpendicular to 'axis'. 'rim' is an example of a parallel curve of 'c0'.

You can use Edit / Show / All to display the RuledSurf that is built between 'c0' and 'rim'.



Procedural Curve.MS2 in <y> view: initial model (left); model with bead 'e0' moved, showing 'moving_point' at another position on the path it sweeps out to form the ProcCurve (right).

Example 2

These mini-examples illustrate how ProcCurves could be used as alternative constructions for other kinds of SurfaceWorks entities:

1 If point is an OffsetPt made from a ring, the ProcCurve would be the same as an OffsetCurv with constant offset. (A graph could be used to vary the offset.)

2 If point is a RelPoint made from a bead or ring, the ProcCurve would be the same as a RelCurve with constant offset — just a translated copy of the basis curve.

3 If point is a MirrPoint, based on a bead, the ProcCurve would be the same as a MirrCurve.

4 If point is an AbsBead at t=.5 on a line joining 'bead1' on 'curve1' and 'bead2' on 'curve2', the ProcCurve would be a curve lying midway between 'curve1' and 'curve2'. This curve would be the same as making the RuledSurf between 'curve1' and 'curve2' and putting a UVSnake on it at v=.5.

See also: ProcSnake, ProcPtSurf, ProcCvSurf

Procedural Snake

Parents and characteristic properties

magnet = the moving point whose path is the ProcSnake
bead/ring/graph1 ... bead/ring/graphN = support(s) for magnet; also define the ProcSnake modeling process

Description

A Procedural Snake is defined by creating one point (magnet) on the ProcSnake, then having SurfaceWorks repeat the magnet creation procedure for all positions of the supporting bead/ring(s) along their host curve(s) or snake(s).

magnet is a magnet or ring constructed directly or indirectly from the 1 or more bead/rings.

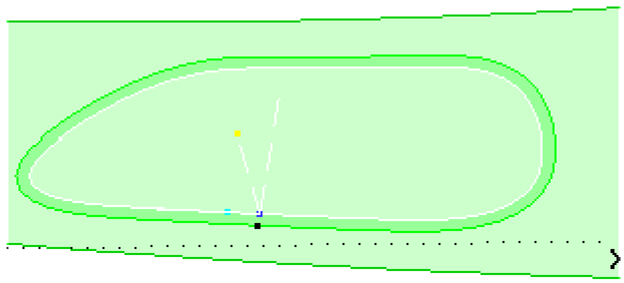
The bead/rings have to be AbsBeads or AbsRings. (not relative to another point)

To generate the ProcSnake, the program varies the parameters of bead/ring1 from t=0 to t=1, varying the other bead/rings in unison, and the ProcSnake is the resulting path of magnet. If the bead/rings don't all have the same t, the program preserves the t offsets between them. Any graphs included as supports are evaluated at each parameter value t and substituted, in order, for any decimal data values in point.

For more information on specifying supports and using graphs, see "Collective Entity Information "

Example 1 ProceduralSnake.ms2

Suppose we have made a closed snake 'portlight' on a hull or cabin side to represent the outline of a portlight, and we want to put a constant-width frame around it.



Procedural Snake.MS2 in <y> view.

'frame' (green) is the ProcSnake in this example. It is made by repeating the procedure of making 'moving_magnet' (red) for every position of 'r0' (blue) along the basis curve 'portlight' (white; a NURBSnake). In this case, moving magnet ('moving_magnet') is only indirectly based on ring ('r0'). Here is the procedure by which 'moving_magnet' is created:

'r0' (blue) is an AbsRing that lies on the NURBSnake 'portlight'. 'tanpt' (cyan) is a TanPoint offset from 'r0' by 0.1 units. 'Pt1' is a Point in the frame created from the 'moving_magnet, the 'tanpt' and the yellow Offset Point. 'moving_magnet' (red; a ProjMagnet) is the projection of 'Pt1' onto the cabin side.

The program generates the ProcSnake by varying the parameter t for 'r0' from 0. to 1. — the resulting path of 'moving_magnet' is the ProcSnake, a snake offset exactly 0.1 units from 'portlight'. You can use Edit / Show / All to display the RuledSurf that is built between 'c0' and 'rim'.

Example 2

These mini-examples illustrate how ProcSnakes could be used as alternative constructions for other kinds of SurfaceWorks entities:

1. If magnet is a ProjMagnet made from a bead, the ProcSnake would be the same as a ProjSnake.

2. If magnet is an RelMagnet made from a bead, the ProcSnake would be the same as a RelSnake with constant offset.

Procedural Surface

Parents and characteristic properties

Moving Entity Type = Point
Moving Point = point. the moving point whose path sweeps out the Procedural Surface
Magnets = magnet. parent(s) for Moving Point ; also define the Procedural Surface modeling process

Moving Entity Type = Curve
Moving Curve = curve. the moving curve whose path sweeps out the Procedural Surface

Beads/Rings = Bead or Ring. parent(s) for Moving Curve; also define the Procedural Surface modeling process

Description

A Moving Entity Type 'Point' Procedural Surface is defined by creating one Point on the Procedural Surface, then having SurfaceWorks repeat the Point creation procedure for all positions of the supporting Magnet(s) over their host surface.

Point is any kind of point entity constructed directly or indirectly from the 1 or more Magnets.

The Magnets have to be Magnets that are not Relative to another entity.

To generate the Procedural Surface, the program varies the parameters of the Magnet(s) over the range [0,1] x [0,1], and the Procedural Surface is the surface swept out by Point. If the Magnets don't all have the same u,v parameters, the program preserves the u,v offsets between them.

A Moving Entity Type 'Curve' Procedural Surface is defined by creating one Curve (a u=constant line) of the Procedural Surface , then having SurfaceWorks repeat the Curve creation procedure for all positions of the supporting Beads/Rings along their host curve(s).

Curve is any curve entity constructed directly or indirectly from the 1 or more Beads/Rings. Curve shows how you want one u=constant line constructed.

The Beads/Rings have to be Beads or Rings that are not Relative to another entity.

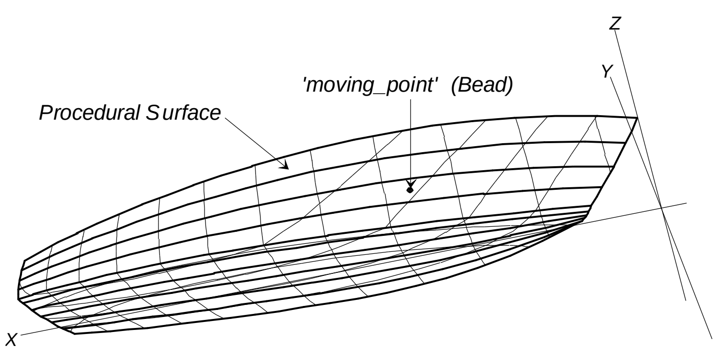
To generate the Procedural Surface , the program varies the parameters of the Beads/Rings from t=0 to t=1, and the Procedural Surface is the surface swept out by Curve. If the Beads/Rings don't all have the same t parameter, the program preserves the t offsets between them.

For more information on specifying parents, see “Understanding SurfaceWorks Entities - Collective Entity Information - Procedural Entities”.

Example: Procpsf1

(The model may take several moments to open. Do not be alarmed — this Procedural Surface just requires significant calculation.)

This hull is a blending between two hulls. 'hull1' on layer1 and 'hull2' on layer2 (to see them, use Tools>Layers to turn their layers on). ‘hull1’ is the Demo hull; ‘hull2’ is a PolySurf (any contiguous surface would do) made from the 'topside' and 'bottom' surfaces on layer2.

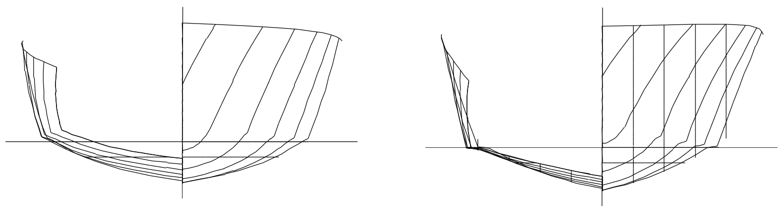


Procpsf1.ms2.

‘hull’ (white) is a Moving Entity Type 'Point' Procedural Surface. Moving Point is a Bead (‘moving_point’) at t=0.5 on a Line 'line1' between ‘mag1’ on ‘hull1’ and ‘mag2’ on ‘hull2’. The Procedural Surface is made by repeating the procedure of making ‘moving_point’ (cyan) for every position of the magnets ‘mag1’ and ‘mag2’ (both red) over their basis surfaces ‘hull1’ and ‘hull2’. In this case, Moving Point (‘moving_point’) is indirectly based on the Magnets (‘mag1’ and ‘mag2’).

The program generates the Procedural Surface ‘hull’ by varying the parameters u,v for ‘mag1’ from 0. to 1.; it varies the parameters for ‘mag2’ in unison with ‘mag1’ — the resulting path of ‘moving_point’ is the Procedural Surface.

By varying the location of 'moving_point' on the Line, you get a family of intermediate hull forms. Fig. 2 body plans show the differences between the initial hull and the hull when the Bead is at t=0.2.



Body plans from Ship Lines for Procpsf1.ms2: initial model with Bead at t=0.5 (left); model with Bead at t=0.2 (right).

Example:

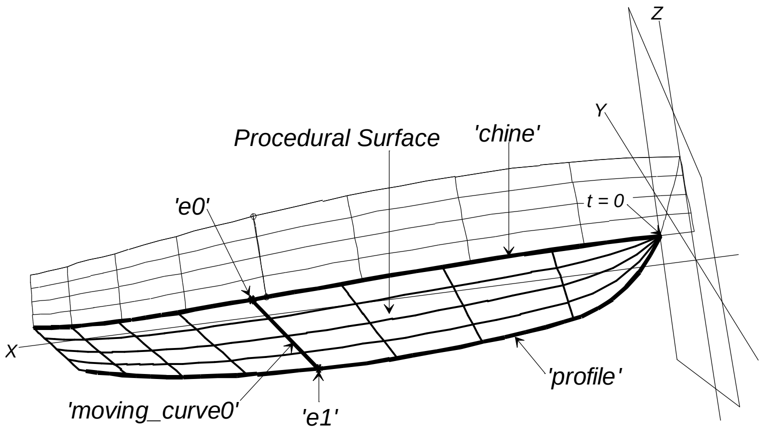
These mini-examples illustrate how Procedural Surfaces could be used as alternative constructions for other kinds of SurfaceWorks entities:

1 If Moving Point is an Offset Point made from a magnet, the Procedural Surface is the same as an Offset Surface with constant offset (a parallel surface).

2 If Moving Point is a Mirrored Point made from a magnet, the Procedural Surface is the same as a Mirrored Surface.

Example: Proccsf2

Chine hulls are frequently designed and constructed with straight lines for sections. This simplifies the framing, since the frames can all be made from straight extrusions such as angle iron. (Note, however, that the resulting surface typically is not developable.) Proccsf2.ms2 contains two procedural surfaces: the side and bottom panels of the hull. We’ll discuss the bottom panel.



Proccsf2.ms2.

‘bottom’ (dark green) is the Procedural Surface in this example. It is made by repeating the procedure of making ‘moving_curve0’ (white) for every position of the bead ‘e0’ (bright green) along its basis curve ‘chine’ (cyan). In this case, Moving Curve (‘moving_curve0’) is directly based on the Beads ‘e0’ and ‘e1’. Here is the procedure by which ‘moving_curve0’ is created:

‘e0’ (bright green) is a Bead at t = 0.5 on the C-spline Curve ‘chine’. ‘e1’ (bright green) is an XYZ Bead on the B-spline Curve ‘profile’ Relative to 'e0' with dX = 0. If you look at the <y> view, you can see that this makes the Line ‘moving_curve0’ (between ‘e0’ and ‘e1’) a straight transverse section for the bottom.

The program generates the Procedural Surface ‘bottom’ by varying the parameter t for ‘e0’ from 0. to 1.; 'e1' follows on 'profile' — the resulting path of ‘moving_curve0’ is the Procedural Surface.

The ‘topside’ surface is similarly constructed with straight frames.

Example:

These mini-examples illustrate how Procedural Surfaces could be used as alternative constructions for other kinds of SurfaceWorks entities:

1 If Moving Curve is a line between two beads at the same t-position on two curves, the Procedural Surface is the same as a Ruled Surface.

2 If Moving Curve is a C-spline Curve made from beads at the same t-position on several master curves, the Procedural Surface is the same as a C-spline Lofted Surface.

Projected Curve

Parents and characteristic properties

Curve = curve entity. parent curve
Mirror = plane, Line or point entity

Description

A Projected Curve is a curve formed by projecting a specified parent Curve onto a Mirror. Each point of Curve is projected normally (i.e., perpendicularly) onto Mirror.

The Projected Curve therefore lies entirely in Mirror. The points of the Projected Curve have the same parameter t values as they had on Curve.

Mirror can be a plane, Line, or point entity:

When Mirror is a plane, each projection line is perpendicular to the plane.
When Mirror is a Line, each projection line is perpendicular to the Line.
When Mirror is a point, each projection line ends at that point.

(Note: Projecting a curve on a point produces a degenerate curve located at that single point. We can think of no utility in doing this, since the point can serve any function required for the Projected Curve.)

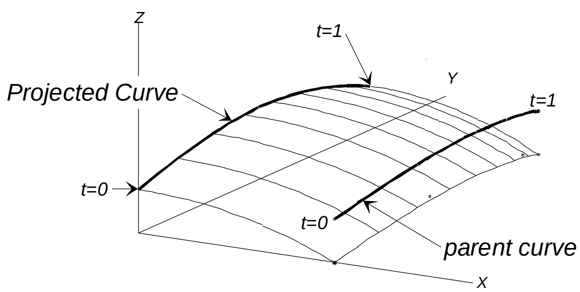
Projected Curves are especially useful for making B-spline Lofted Surfaces end accurately perpendicular to a plane.

For comparative information about projected and mirrored entities, see "Understanding SurfaceWorks Entities - Collective Entity Information - Projected and Mirrored Entities".

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is distributed the same as on the basis curve.

Example: ProjCurve

In this example, Relative Curve 'curve2' (the parent curve) has been projected onto the *X = 0 plane to create the Projected Curve 'curve3'. 'curve1', 'curve2', and 'curve3' are then used to create the Arc Lofted Surface 'surface1'. Since 'curve3' lies in the *X = 0 plane and the Arc Lofted Surface is of Type: tangent at curve3 end, this ensures that the 'curve3' edge of 'surface1' is perpendicular to the *X = 0 centerplane.



ProjCurve

See also: Mirrored Curve, Projected Magnet, Projected Point, Projected Snake

Projected Magnet

Parents and characteristic properties

Point = point entity
Magnet/Surface = magnet, ring or surface. designates the surface the Projected Magnet will lie on
Mirror/Surface = plane, surface, Line or point entity

Description

A Projected Magnet is the projection of a Point along a straight line onto a surface (Magnet/Surface). The projection can be done across any valid type of Mirror/Surface:

When Mirror/Surface is a plane, the projection line is normal (perpendicular) to the plane.
When Mirror/Surface is a line, the projection line is normal (perpendicular) to the line.
When Mirror/Surface is a point, the projection line radiates from the point.
When Mirror/Surface is a surface (it must be the same surface as Magnet/Surface, the Projected Magnet is to lie on), the projection is along the local normal direction to the surface; the Projected Magnet will be along the line from Point to the nearest point of the surface to Point.

Magnet/Surface can be either a magnet (including rings) or a surface:

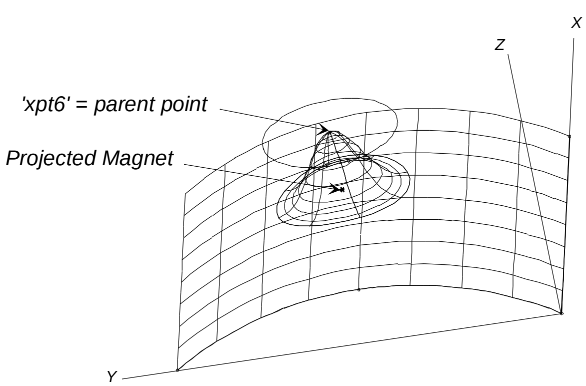
When Magnet/Surface is a magnet, it serves two purposes:

- It specifies which surface the Projected Magnet is to lie on: the surface that is parent to the magnet.
- In case of multiple intersections of the surface with the projection line, the Projected Magnet will be made from the intersection nearest to the magnet.

When Magnet/Surface is a surface, it specifies directly which surface the Projected Magnet is to lie on. In case of multiple intersections of the surface with the projection line, the Projected Magnet will be made from the intersection closest to u= 0.5, v = 0.5.

Example: ProjMagnet

This model has a style-7 Blister Surface ‘blister’ (blue) attached to a Translation Surface (green). The Projected Magnet ‘projmag’ (red’) is at the center of the base of the Blister. It is the projection of ‘xpt6’ (dark magenta) onto the Translation Surface. It serves as the Snake2 parent of the Blister.



ProjMagnet

See also: Magnet, Projected Point, Projected Curve, Projected Snake

Projected Point

Parents and characteristic properties

Point = point entity. parent point
Mirror = plane, Line or point entity

Description

A Projected Point is made by specifying the parent Point and a Mirror of projection. Point is projected normally (i.e., perpendicularly) onto Mirror. The resulting Projected Point lies in Mirror and is in fact the closest point in the mirror to the parent Point.

Mirror can be a plane, Line, or point entity:

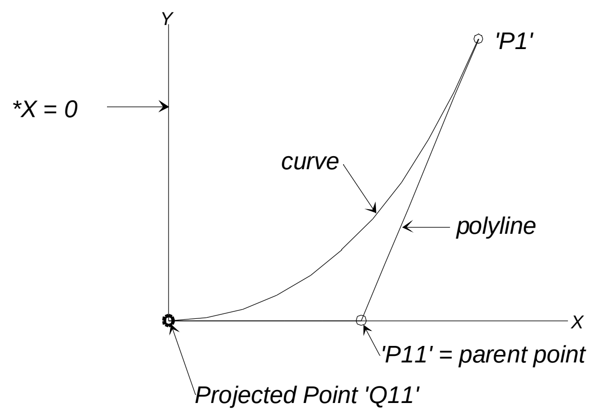
When Mirror is a plane, the projection line is perpendicular to the plane.
When Mirror is a Line, the projection line is perpendicular to the Line.
When Mirror is a point, the projection line ends at that point.

(Note: We can think of no utility in projecting a point on a point, since this just results in a new point entity at a place where you already have one.)

You are not permitted to edit coordinates of a Projected Point, or drag it, like you can with most point entities. This is because it takes its position from that of its parents, Point and Mirror.

For comparative information about projected and mirrored entities, see "Understanding SurfaceWorks Entities - Collective Entity Information - Projected and Mirrored Entities".

Example: ProjPoint



ProjPoint

Projected Points are particularly useful for making B-spline Curves end perpendicular to a given plane, for example:

'P1' and 'P11' are Points and 'Q11' (dark cyan) is the projection of 'P11' (red) onto *X = 0.

'curve' (cyan) is a degree-2 B-spline Curve formed by the 3 control points 'P1', 'P11', and 'Q11'. Visibility for 'curve' has the polyline turned on. Since 'Q11' is the perpendicular projection of 'P11', the polyline segment joining 'P11' and 'Q11' is perpendicular to 'centerplane'. Since a B-spline Curve is tangent to its polyline at each end, 'curve' ends perpendicularly at *X = 0.

See also: Point, Offset Point, Mirrored Point, Projected Curve, Projected Magnet, Projected Snake, B-spline Curve

Projected Snake

Parents and characteristic properties

Curve = curve entity
Magnet/Surface = magnet, ring or surface. designates the surface on which the Projected Snake will lie
Mirror/Surface = plane, surface, Line or point entity
Draft while projecting (when checked; only available if Mirror / Surface is a plane entity) = draft angle in degrees and whether to draft outward or inward

Description

A Projected Snake is the projection of Curve onto a Surface. Each point of the curve is projected along a straight line to locate the corresponding point on the Projected Snake. The projection can be done using any valid type of Mirror/Surface:

When Mirror/Surface is a plane, the projection lines are normal (perpendicular) to the plane. Note: If you want to specify a draft angle, Mirror/Surface must be a plane and in this case the projection lines are at an angle, not normal to the plane.
When Mirror/Surface is a Line, the projection lines radiate from the axis line and are normal (perpendicular) to the Line.
When Mirror/Surface is a point, the projection lines radiate from the point.
When Mirror/Surface is a surface (it must be the same surface the Projected Snake is to lie on), the projection is along the local normal direction to Surface; each point of the Projected Snake will be at the foot of a line from the corresponding point on Curve dropped normally to the Surface (usually this is the closest point on the surface).

Magnet/Surface specifies the surface the Projected Snake will lie on. It can be either a magnet (including rings) or a surface:

When Magnet/Surface is a magnet or ring, it serves three purposes:

- It specifies which surface the Projected Snake is to lie on: the surface that is parent to Magnet.
- In case of multiple intersections of the surface with the projection lines, the Projected Snake will be made from the intersection whose t=0 end is nearest to Magnet.
- It provides a starting point for the search procedure SurfaceWorks uses to locate the first point (t=0) of the Projected Snake.

Whenever you create a Projected Snake, we recommend you use a magnet or ring for Magnet/Surface, locating it reasonably near where the Projected Snake will begin.

When Magnet/Surface is a surface, it tells directly which surface the Projected Snake is to lie on. In case of multiple intersections of Surface with the projection lines, the Projected Snake will be made from the intersection whose t=0 end is nearest to u=0.5, v=0.5. The search for an intersection for the t=0 end will begin at u=0.5, v=0.5.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t's distribution corresponds to the labeling of the basis curve.

Draft

Note: In the dialog, the draft specifications will be grayed out unless Mirror/Surface is a plane.

To make a molded part easier to remove from a mold, you can use a Projected Snake with a draft angle as one of the parents for a tapered surface. The plane specified for Mirror/Surface will be used as the neutral plane for the draft.

To specify a draft angle:

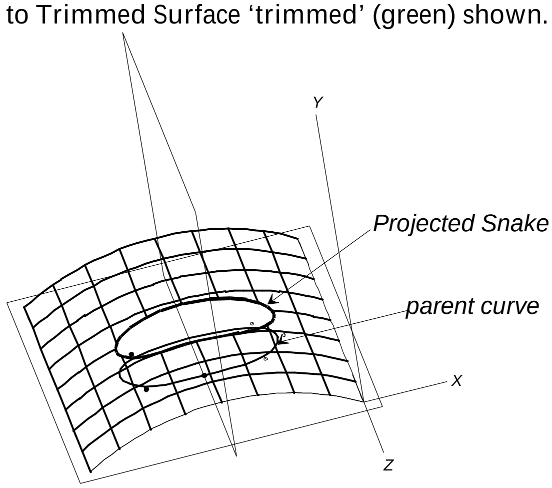
If you want to specify a draft angle, check the “Draft While Projecting” box, then enter the draft angle in degrees.

If you want the Projected Snake to be slanted in the opposite direction, check the “Reverse draft” box.

For an example, see Example 2 below.

Example 1: ProjSnake

Projected Snake ‘projsnake’ (blue) is used to define the hole in the Trimmed Surface. Its parent Curve ‘parent’ (cyan) has been projected onto Magnet/Surface ‘surface1’ (gray) using the ‘xplane2’ plane as its Mirror/Surface. ‘surface1’ is the parent surface to Trimmed Surface ‘trimmed’ (green) shown.



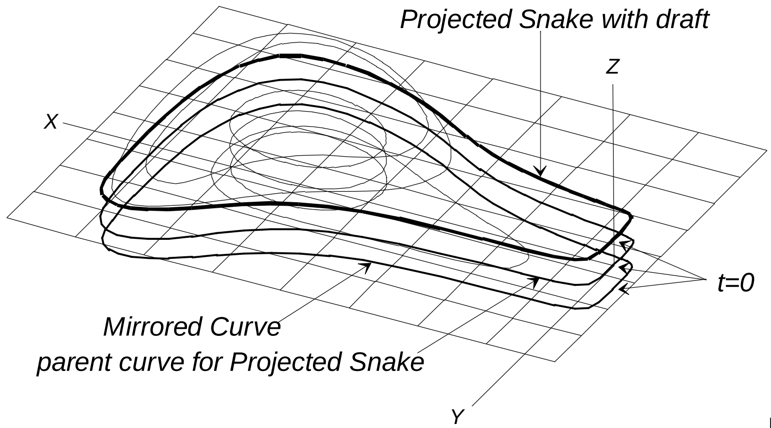
ProjSnake

Example 2: ProjSnake-Draft

This scissors handle model uses a Projected Snake with a draft angle to facilitate removal from the mold.

‘seam’ (blue) runs around the outside of the handle and defines the outer seam between the two mold halves. ‘draft’ (magenta) is a Projected Snake of ‘seam’ onto the flat surface ‘surface3’ (gray), using ‘*Z=0’ (the internally-defined Z=0 plane) for Mirror/Surface. The draft angle is 3 degrees. ‘surface1’, the lower handle surface (in this view), is a B-spline Lofted Surface that uses ‘seam’ and ‘draft’ for its first two control curves.

‘mirrored’ is a Mirrored Curve of ‘draft’ across the Z=0 plane. ‘surface2’, the upper handle surface (in this view), is a B-spline Lofted Surface that uses ‘seam’ and ‘mirrored’ as its first two control curves.



ProjSnake-Draft

See also: Projected Point, Projected Magnet, Projected Curve

Projected Surface

Parents and characteristic properties

surface = basis surface
mirror = a plane, Line, or point object

Description

A Projected Surface is a surface formed by projecting a specified surface (called the "basis surface") onto a mirror. Each point of surface is projected normally (i.e., perpendicularly) onto mirror. The Projected Surface therefore lies entirely in mirror. If the basis surface is later changed, the Projected Surface will be automatically relocated to remain in the perpendicularly-projected position.

mirror can be a plane, Line, or point object:

When mirror is a plane, each projection line is perpendicular to the plane.

When mirror is a Line, each projection line is perpendicular to the Line.

When mirror is a point, each projection line ends at that point.

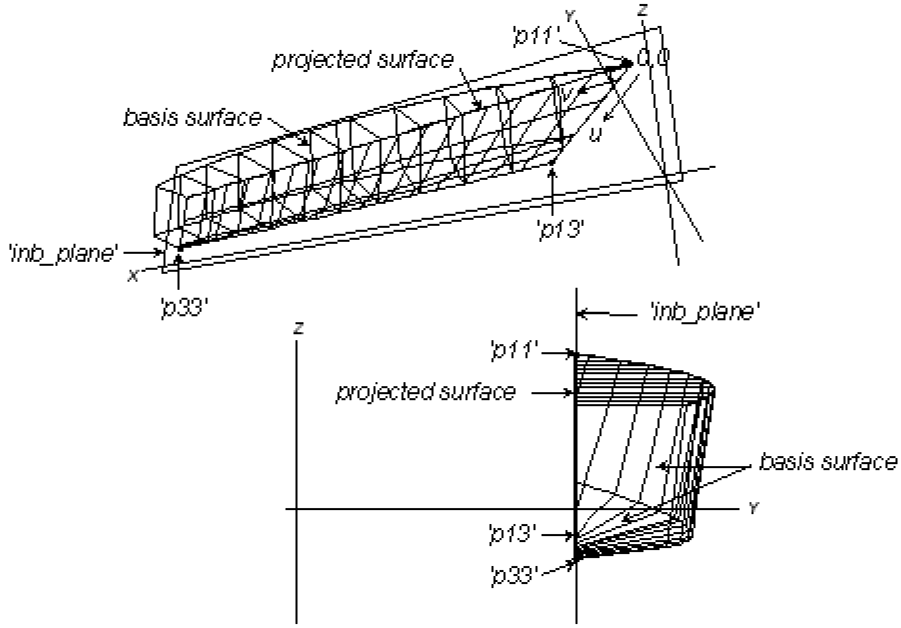
(Note: When mirror is a point, each projection line passes through that point. This Projected Surface would be doubly degenerate, i.e. the entire surface would be at a single point. We do not know of any present utility in making these degenerate surfaces.)

Example projsurf.ms2

'outboard' (the basis surface) is the outboard surface for the starboard hull of a planing catamaran design. It is like half of a conventional V-bottom hull. 'inb_plane' is a plane defined by three points: 'p11' at the top of 'MC1' (the point of the stem), 'p13' at the bottom of 'MC1', and 'p33' at the bottom of 'MC3'. (Since its three control points are all at Y = 10, 'inb_plane' currently is the same as the YPlane at Y = 0.)

'inboard' (the projected surface) is the inboard surface for the starboard catamaran hull, obtained by projecting 'outboard' onto 'inb_plane. 'inboard' is flat and parallel to the Y = 0 plane (the centerplane), at Y = 10.

Note that we could create the port hull for this catamaran by turning on Y-symmetry (Settings / Model), as we did in the “MirrSurf” entity description.



PROJSURF.MS2 at Lat -20, Lon 60 (top) and <x> view (bottom).

See also: Mirrored Surface, Projected Point, Projected Curve

Proximity Points

Characteristic Data

Proximity Bead

type (kind), 0 | 1 = minimum / maximum signed distance
type (kind), 0 | 1 = unconstrained / constrained
bead/curve
mirror/surface

Proximity Ring

type (kind), 0 | 1 = minimum / maximum signed distance
type (kind), 0 | 1 = unconstrained / constrained
bead/curve
mirror/surface

Proximity Magnet

type (kind), 0 | 1 = minimum / maximum signed distance
type (kind), 0 | 1 = unconstrained / constrained
bead/curve
mirror/surface

Description

The ProxBead, ProxRing and ProxMagnet entity types solve the following problems:

ProxBead (ProxRing):

(Type 0) Find the point on a curve (snake) that has the minimum signed distance from a mirror / surface.

(Type 1) Find the point on a curve (snake) that has the maximum signed distance from a mirror / surface.

ProxMagnet

(Type 0) Find the point on a surface that has the minimum signed distance from a mirror / surface.

(Type 1) Find the point on a surface that has the maximum signed distance from a mirror / surface.

"Signed distance" is a measure of the position of a point in relation to a mirror or surface, that takes the orientation of the mirror or surface into account. In the case of a point or line mirror, signed distance is always positive, and increases outward from the mirror. In the case of a plane mirror, signed distance is zero on the plane, positive on one side and negative on the other. In the case of a surface mirror, signed distance is zero on the surface, positive in the direction of the positive normal and negative in the other direction.

Constrained/ Unconstrained

There is an issue of whether the ProxBead is constrained to the interval [0,1], or is free to go onto the extensions of the curve; similarly for a ProxMagnet, whether it can use the extension of the surface. Both alternatives appear useful in different circumstances, so both are provided by means of a second type: 0 for unconstrained, 1 for constrained.If a bead (ring, magnet) is given for the bead / curve (ring / snake, magnet / surface), SurfaceWorks uses it for the starting location, and searches for a nearby local minimum or maximum. Otherwise, SurfaceWorks starts with a tabulation of the host curve (snake, surface) and identifies the closest / farthest point in the table as a startinglocation for the search.

Associated errors

(1) Failed to converge.

(2) No solution exists. This will be a fairly common occurrence; say, from choosing the wrong type. On a sailboat hull (unconstrained), the maximum Y point will exist and be well defined, but in general there is no minimum Y -- the extended surface goes on to minus infinity. The optimization will be trying to run to minus infinity, and we have to detect that and stop it before the numbers get outrageous.

Sample files

ProxBead1.ms2 This model contains 4 ProxBeads hosted on a circle. 'e1' is at the closest point on the circle to Point 'p3', and 'e2' is the point farthest away from 'p3'. 'e1' and 'e2' both use the circle as parent, so do not specify a starting point. 'e3' is the point closest to Point 'p4', and 'e4' is the point farthest away from Point 'p5'. 'e3' and 'e4' both use AbsBead 'e0' as a starting point.

ProxBead3.ms2 This model contains 3 ProxBeads hosted on a semicircle, each constrained to the 0-to-1 parameter interval (their second type is 1). 'e1' is the point on 'c0' that is closest to Point 'p3'; 'e2' is the point that is farthest away from 'p3'. 'e3' is also at a local maximum distance from 'p3'; its parent is AbsBead 'e0', providing a starting position for the search, which therefore finds the local rather than global maximum. If 'e0' is moved to the opposite side of 'e1', then 'e3' jumps to the t=0 end of 'c0'.

ProxRing1.ms2 This model has 2 ProxRings hosted on a BSnake. Each is at the minimum distance from a magnet. 'r1' uses the snake as parent; 'r2' uses AbsRing 'r0', which gives it a starting location for the search.

ProxMagnet1.ms2 This model has a single ProxMagnet 'm1', used to locate the maximum beam location on a boat hull. The mirror is the predefined plane *Y=0, whose positive orientation is toward positive Y. Therefore the ProxMagnet's first type is 1, seeking a maximum signed distance.

MaxBeam.ms2 In this model a construction, using a Proximity Ring, is set up to find a curve showing the maximum beam on a vessel's topsides. The Proximity Ring 'Prox1' draws out a Procedural Snake caused by the action of moving its parent snake from sheer to chine. The Parents of the Procedural Snake are the Proximity Ring and the Ring 'MovingPoint', which is the parent of the ProxRing's host UV Snake.

Radius Arc

Parents and characteristic properties

point1, poin2, point3 = control points
radius = radius of arc
type = kind of radius arc

Description

The RadiusArc is a circular arc of a specified radius. It is defined by three control points and a radius, and can be built in several different ways, as designated by type.

radius is a decimal number, positive or negative. For type-1 and type-2, radius must be less in absolute value than twice the distance from point1 to point3.

type specifies how the radius arc is built from the control points:

type-1 The arc runs from point1 to point3, in the plane of the 3 points; type-1 is the <u>shorter</u> arc, with <u>angle < 180 deg</u>.

type-2 The arc runs from point1 to point3, in the plane of the 3 points; type-2 is the <u>longer</u> arc, with <u>angle > 180 deg</u>.

type-3 This arc rounds the corner formed at point2 by the directions to point1 and point3. Its t=0 end is tangent to the line from point1 to point2; its t=1 end is tangent to the line from point2 to point3.

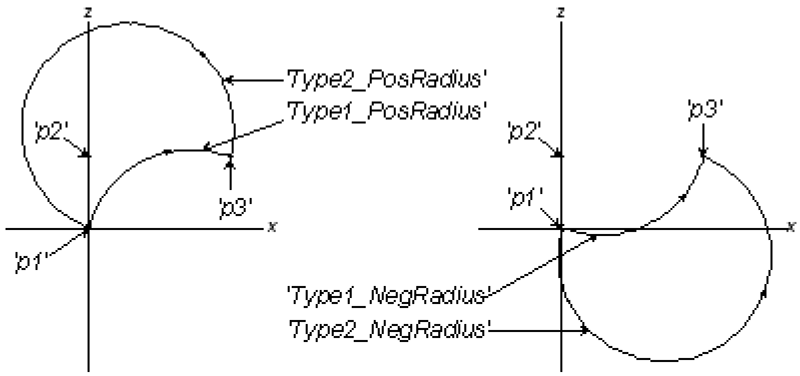
For type-1 and type-2 RadiusArcs, note that in a given plane, there are four possible arcs of a given radius between two points:

type-1, positive radius

type-1, negative radius

type-2, positive radius

type-2, negative radius



RadiusArc types 1 and 2: four possible arcs between two points.

The program's default relabel produces uniform labeling with respect to arc length or angle.

Example 1 radiusarc1.ms2

Example: radiusarc1.ms2

This is an example of a constant radius deck made by sweeping a RadiusArc to generate a ProcCvSurf (Procedural Curve Surface).

'deck_radius' (red) is a type-1 RadiusArc. It has a radius of 20 and is built between:

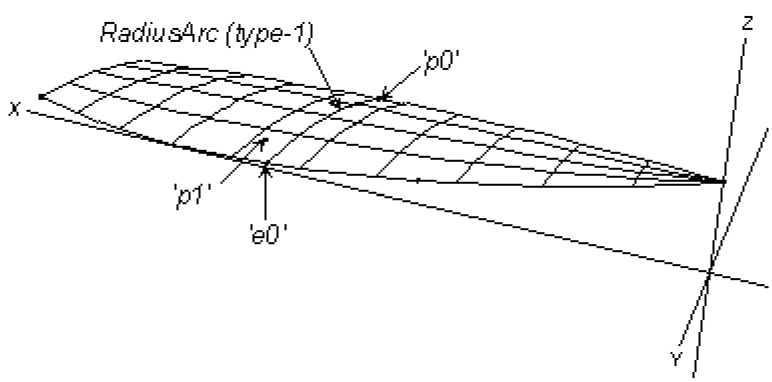
'e0' (bright green; an AbsBead on 'sheer') = point1 for the arc

and

'p0' (cyan; a MirrPoint of 'e0' across the centerplane) = point3 for the arc.

'p1' (yellow; a RelPoint off 'e0', at dX=0, dY=0, dZ= .1) is point2 for the arc. It orients the arc in a plane parallel to the X=0 plane.

The deck is a ProcCvSurf for which the moving curve is 'deck_radius' and bead = 'e0'. To generate the ProcCvSurf, SurfaceWorks varies t for 'e0' from 0 to 1, and the ProcCvSurf is the surface swept out by 'deck_radius'.



RADIUSARC1.MS2 at Lat -20, Lon 30.

Example 2 radiusarc3-polycurve2.ms2

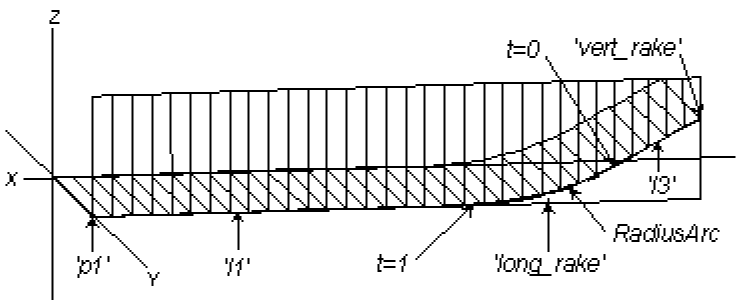
Example 2: radiusarc3-polycurve2.ms2

This model is a hard-chine barge with raked ends. A type-3 RadiusArc is used to form the 20-foot-radius curve in the chine.

‘radiusarc’ (bright green) is a type-3 RadiusArc made from ‘vert_rake’ (blue, a bead on ‘l2’), ‘long_rake’ (blue, a bead on ‘l0’), and ‘p2’ (cyan, a ProjPoint of a point at the origin). ‘radiusarc’ is tangent to ‘l3’ (yellow) at its t=0 end and to ‘l1’ (magenta) at its t=1 end.

‘chine’ (cyan) is a PolyCurve2 made from ‘l1’, ‘radiusarc’, and ‘l0’.

You can vary the longitudinal and vertical extents of the rake by dragging ‘long_rake’ and vert_rake’ respectively. The RadiusArc adjusts itself to maintain its specified radius of 20 ft and it remains tangent to the lines ‘l1’ and ‘l3’.



RADIUSARC3-POLYCURVE2.MS2 at Lat 10, Lon 100.

For discussion of the PolyCurve2 in this model, see the end of the first PolyCurve2 Entity Description example.

See also: Arc

Relabel

Characteristic data

Degree = type of B-spline used in the relabeling graph
Values = relabel values

Description

A Relabel entity controls the relabeling (re-parameterization) of a curve, snake or some surfaces. It is not a visible entity in itself, so it has no color and visibility, but its effects on labeling can be made visible by showing tickmarks for the affected curve or snake (see "Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Visibility"). Its effect on surfaces is harder to see, but magnets will change their absolute location on a surface as the surface's u-v parameterization changes.

Relabeling does not affect the 3D shape of a curve, snake or surface, but it does affect the display of the curve, snake or surface with a finite number of points. Relabeling can affect the 3D shape of surfaces built from the curve or snake.

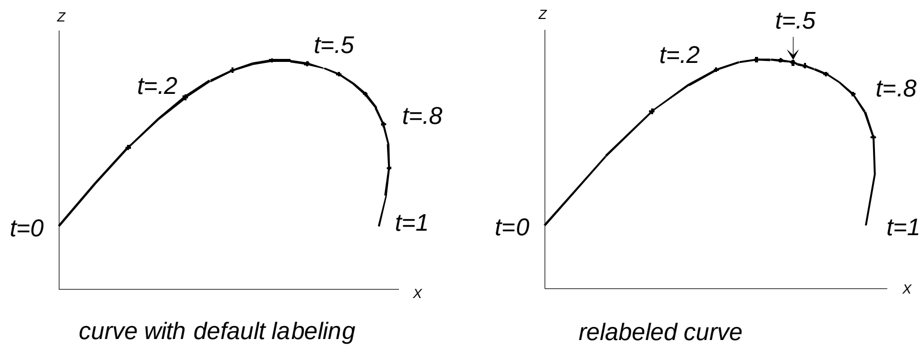
Degree is a positive integer specifying the B-spline type used in the relabeling graph: 1 = linear, 2 = quadratic, 3 = cubic, etc. The higher the type, the stiffer the B-spline.

Values are edited by clicking on the "Edit Values" button and using the "Relabel Values" dialog box.

For more details about specifying Relabel entities, see "Editing Models - Relabeling Curves and Snakes".

Examples: relabel and relabel3

In example 1, the B-spline Curve 'curve' is labeled with the default relabel ('*'). For a B-spline Curve, this natural labeling concentrates t in the more highly curved areas of the curve (left-hand figure).



relabel (left) and relabel3 (right) A B-spline Curve with the default labeling (left) and relabeled with Example 2 'relabel3' (right).

In example 2, the B-spline Curve 'curve' is relabeled with the Relabel entity 'relabel3'. Type for 'relabel3' is 2, which means the B-splines used in the relabeling formulas are type-2. The set of relabeling values { 0. .45 .55 1. } concentrates t in the middle of the curve (right-hand figure).

Relative Curve

Parents and characteristic properties

Curve = curve entity
Point1 and Point2 = point entity. offsets at ends of Curve

Description

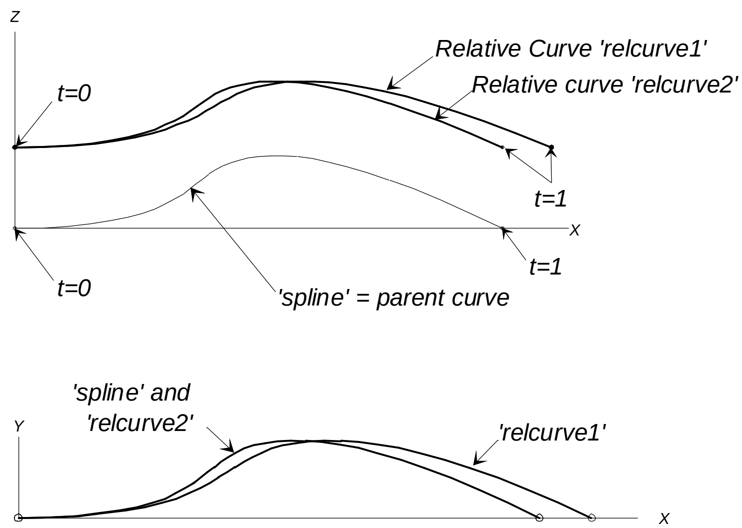
A Relative Curve is a curve formed by offsetting points from a parent curve. You must specify the parent Curve and two points, which become the start (Point1) and end (Point2) points for the Relative Curve.

The distance and direction of Point1 from the start of Curve establishes an offset for the start (t = 0), and the distance and direction of Point2 from the end of Curve establishes an offset for the end (t = 1).

To make a constant offset, make the offset at the two ends the same for all 3 coordinates. Using two relative Points, one based on each end of the parent curve, is a convenient way to do this.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t is distributed the same as for the parent curve.

Example: RelCurve3



RelCurve3: Home view (top); Bottom view rotated 180 degrees (bottom).

‘spline’ (magenta) is the parent curve. The other two curves are Relative Curves, each beginning at the same point in space but ending at different points. Points ‘q1’ and ‘q2’ define the two ends of ‘relcurve1’ (green). Since the two endpoints are not equally offset from the ends of the parent curve, this curve is not at a constant offset from the parent curve. The relative Points ‘rp1’ and ‘rp2’ define the two ends of ‘relcurve2’ (cyan). This curve is at a constant offset from the parent curve.

Relative Snake

Parents and characteristic properties

snake = basis snake
magnet1 and magnet2 = offsets at ends of snake
graph = name of a BGraph object or ‘*’ for default graph

Description A Relative Snake is a snake formed by offsetting points from another snake, called the “basis snake”. You have to name the basis snake and two magnets, which become the start (magnet1) and end (magnet2) points for the Relative Snake. The basis snake and the two magnets must all lie in the same surface.

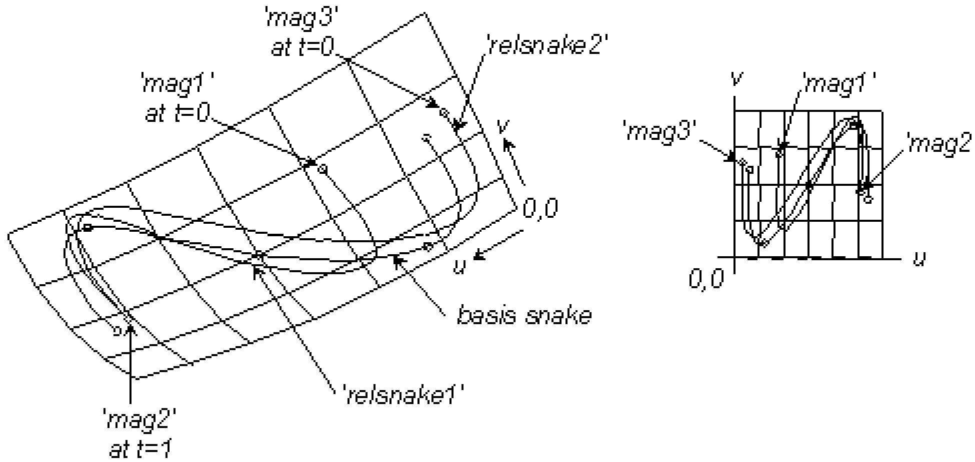
The distance and direction of magnet1 from the start of snake in the u-v parameter plane establishes an offset for the start (t = 0), and the distance and direction of magnet2 from the end of snake establishes an offset for the end (t = 1).

The effect of the graph is similar to its action in RelCurve: it governs how the end displacements are combined and distributed in determining the displacement of each point on the RelSnake from the corresponding point on snake.

To make a constant offset, make the offset at the two ends the same in both u and v coordinates. The use of two Relative Magnets, one based on each end of the basis snake, is a convenient way to do this.

relabel is used to relabel the curve. The program’s default relabel ‘*’ produces the “natural” labeling, in which the parameter t is distributed the same as for the basis curve, in the u-v parameter space.

Example relsnk1.ms2



RELSNK1.MS2 at Lat -60, Lon 60.

'csnake', a snake lying in the surface 'patch', is the basis snake for the two relative snakes 'relsnake1' and 'relsnake2', both also lying in 'patch'. 'relsnake1' begins at 'mag1' and ends at 'mag2'. 'relsnake2' also ends at 'mag2', but it begins at 'mag3'. Since its ends are equally offset from the ends of the basis snake, 'relsnake2' is at a constant offset from

'csnake' in the u-v plane. (In general, this will not be a constant offset on the surface.)

Relative Surface

Parents and characteristic properties

surface = basis surface
point1, (point2, point3, point4) = offsets at corners of surface (1 or 4 required)

Description

A Relative Surface is formed by offsetting the points of the basis surface, in a way similar to making a Relative Curve from a basis curve. You specify a Relative Surface by naming the basis surface and new positions for either:

one corner (point1) — Each point of the RelSurf is offset from the corresponding point of the basis surface by a constant distance and direction — the offset of point1 from the u=0, v=0 corner of the basis surface. The u=0, v=0 corner of the RelSurf is at point1.

Note that a RelSurf using a constant offset is NOT the same as creating an OffsetSurf (in which the offsets are applied perpendicularly to the basis surface). The perpendicular distance between a RelSurf and its basis surface would not in general be constant.

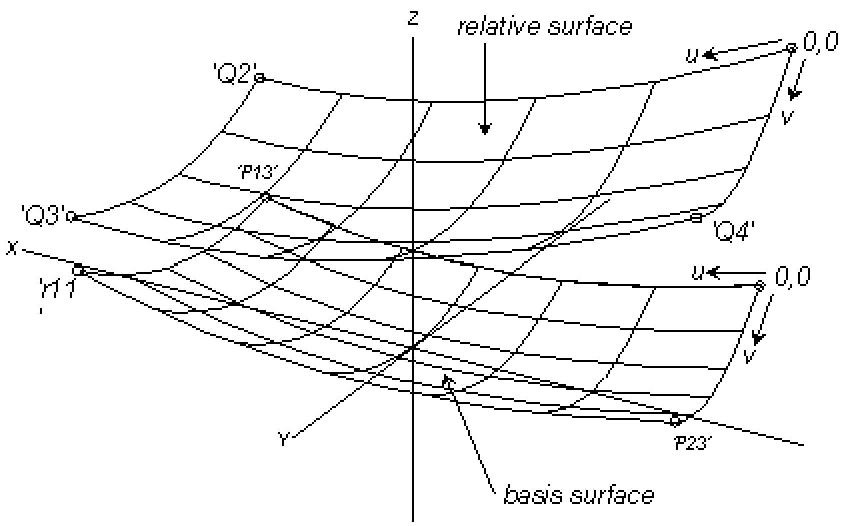
each of the four corners (point1 to point4) — The surface is stretched out to meet these four corner points. Intermediate points are offset by amounts linearly related to their u, v coordinates.

The order of naming the four corner points is important. The corners are identified as follows:

point1 is at u = 0, v = 0 point3 is at u = 1, v = 1 point2 is at u = 1, v = 0 point4 is at u = 0, v = 1

For most accurate results, give the RelSurf the same divisions as the basis surface.

Example relsurf2.ms2



RELSURF2.MS2 at Lat -20, Lon 60.

‘patch’ is the basis surface for the relative surface ‘relsurf’. ‘Q1’, ‘Q2’, ‘Q3’, and ‘Q4’ are the four corners of ‘relsurf’. They are RelPoints offset from the four corner points of ‘patch’. ‘r11’ is an AbsMagnet at the u = 1, v = 1 corner of ‘patch’. The translation surface has no point object defined at that corner, yet one is needed in order to define the offset of the relative surface from that corner, hence the need for ‘r11’.

Note that in this example the surface divisions are not coordinated: ‘patch’ has 6x3 5x2, while ‘relsurf’ has 5x3 4x2 — we did this just to make the two surfaces easier to identify in black-and-white figures.

See also: Offset Surface, Relative Curve, Relative Snake

Revolution Surface

Parents and characteristic properties

Meridian curve = curve entity
Axis line = Line. axis of rotation
Angle1 and Angle 2 = starting and ending angles of rotation

Description

A Revolution Surface is formed by rotating a copy of the parent Meridian curve about an axis. You specify the surface by naming the Meridian curve, the Axis line, and the starting and ending angles of rotation. The angles of rotation are measured from the original position of the parent curve. Meridian curve in its “start” position (at Angle1) forms one edge of the surface, where u = 0, and the final position (at Angle2) of the curve is u = 1. The u parameter on the surface is the same as the t parameter of the parent curve. The sense of rotation is determined by a right-hand rule: if you grasp the axis with your right hand, with the thumb pointing from point1 towards point2, then your fingers point in the direction of positive angle. The angle is measured in degrees. A complete surface of revolution can be generated by specifying Angle1 = 0 and Angle2 = 360.

Relabel is used to control the labeling of the v-direction of the surface.

Example 1: RevSurf1-Line

The three-quarter sphere below is the Revolution Surface ‘surface1’. Its Meridian curve ‘xcurve1’ (magenta) is a parent Arc, and its Axis line ‘axis’ (brown) was created from the two parent points ‘xpt1’ and ‘xpt2’. ‘surface1’ starts with ‘xcurve1’ at Angle1 = -180 degrees and ends with it at Angle2 = 90 degrees, producing the three-quarter sphere.

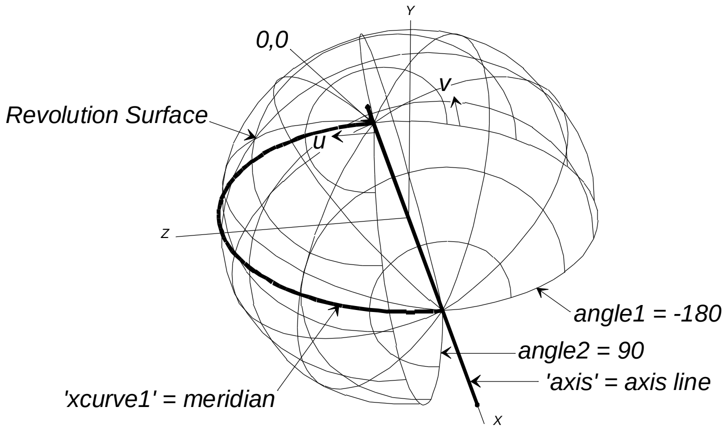


Fig. 1. RevSurf1-Line

Example 2: RevSurf2

‘meridian’ (cyan) is a B-spline Curve made from the 4 control points ‘p1’ to ‘p4’. Points ‘P’ and ‘Q’ define the Line ‘axis’ (magenta) used as the Axis line for the revolution. The Revolution Surface ‘shell’ starts with ‘meridian’ at 0 degrees and ends with ‘meridian’ at 135 degrees (upper figure).

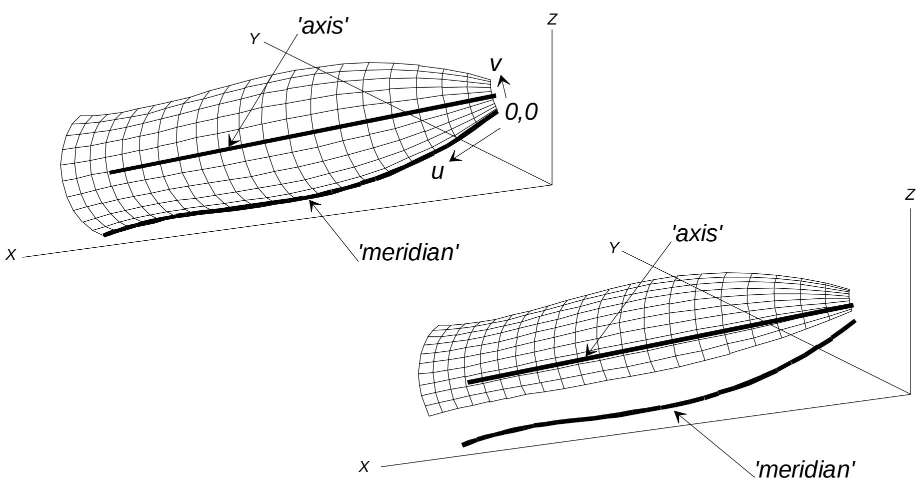


Fig. 2. RevSurf2: angle1 = 0 (top); angle1 = 45 (bottom).

The lower figure shows a surface made from the same parent curve and axis, but with ‘meridian’ sweeping the surface beginning at 45 degrees (and ending at 135, same as before).

Ring

Parents and characteristic properties

Snake = parent snake for Ring
t (or dt when Relative to is checked) = location along snake
Relative to (when checked) = the parent ring for relative location

Description

A ring is a point constrained to lie on a snake. A Ring is specified by naming the parent Snake and giving a parameter value t along that snake (or dt offset from another ring on the same snake). A Ring can be used wherever a point, bead, magnet, or ring is called for.

Snake can be any kind of snake. When Relative to is checked, the Snake field will gray out — the ring you choose to offset from will suffice to identify the snake on which the relative Ring will lie.

Relative to: If you want to make the Ring relative to (offset from) another ring, which must be on the same snake, check the box and specify the parent ring. If you change the parent ring, or if you change between absolute / relative coordinates, SurfaceWorks automatically will adjust the location value to maintain the Ring’s position.

The default location for a Ring depends on whether it has absolute or relative coordinates:

> When it has absolute coordinates, the default location of the new Ring is at t = 0.5 of the parent snake.
>
> When it has relative coordinates (“Relative to” is checked), the default location of the new Ring is dt = 0, which puts it at the location of the “Relative to” ring.

Example: Rings

’uvsnake’ (magenta) is a u=constant UVSnake on the domed surface ‘surface1’ (dark blue). ‘ring’ (bright blue) is a Ring on ‘uvsnake’, located at t = .877. ‘relring’ (also bright blue) is a relative Ring based on ‘ring’, at dt = .029 (Fig. 1). ‘ring’ and ‘relring’ are the start and center points for the Arc ‘circle’, which is projected onto ‘surface1’ to form the Projected Snake ‘base’, which accurately joins the handle (‘surface2’; green) to the dome surface.

‘ring’ can be moved back and forth along ‘uvsnake’ and ‘relring’ will maintain its offset to it (Fig. 2).

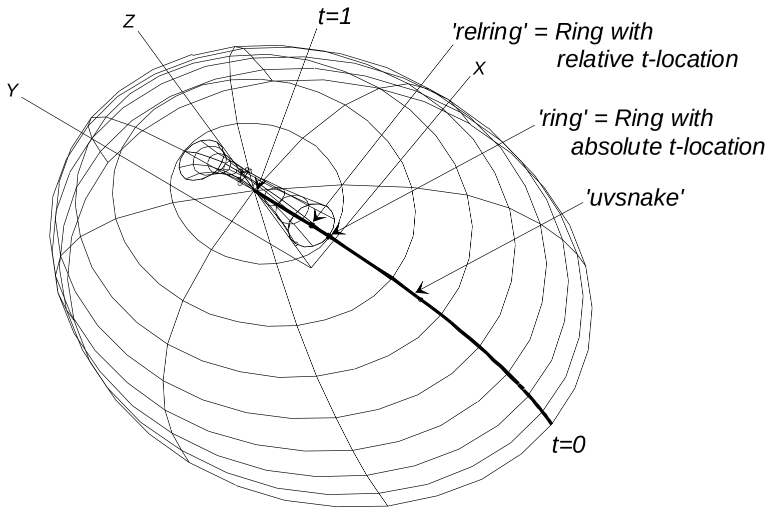


Fig. 1. Rings: initial.

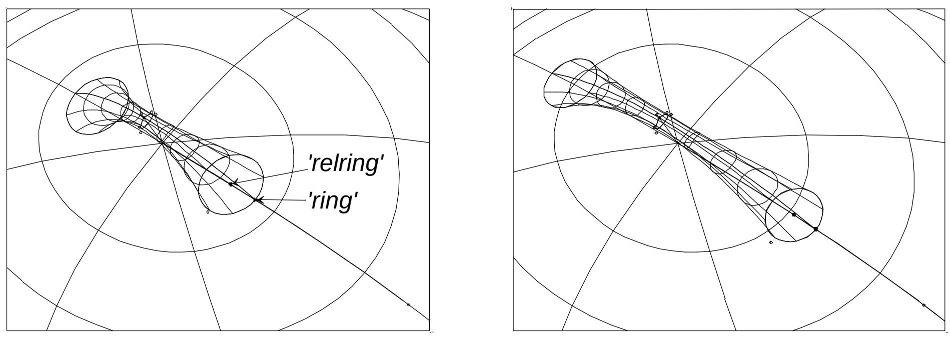


Fig. 2. Rings: with ‘ring’ moved to t = .897 (left) and t = .840 (right).

See Also: Bead, Arc-Length Bead, Arc-Length Ring, XYZ Bead, XYZ Ring

Rolling Ball Fillet

Parents and characteristic properties

Type = fillet type (see below)
Curve = specifies location of fillet
Magnet/surface1 = first surface to join with fillet
Magnet/surface2 = second surface to join with fillet
Radius = radius of fillet
Graph = used to modify Radius

Characteristic data

A rolling-ball fillet is the envelope of a continuous family of spheres that contact two surfaces. There is primary interest in rolling-ball fillets of constant radius, because these can be machined with a ball-end cutter of the same radius. There is secondary interest in rolling-ball fillets of variable radius.

Type tells which of 4 possible fillets you want to construct along this intersection:

0 = on the side of positive normal on both surfaces

1 = negative normal on surface1, positive on surface2

2 = negative normal on surface2, positive on surface1

3 = on the side of negative normal on both surfaces

(4-7) = These all make full tubes; the type selects the quadrant based on positive or negative normal directions for the two surfaces, and would just be the fillet type, plus 4. The tube should be generated like a SweepSurf that uses * for the default guide, and a full circle for the 'shape' curve.

radius = dimensional value (length units)

graph = unitless multiplier of *radius*. Default graph * is constant = 1.

The Curve serves only to specify a family of planes in which the construction occurs; the planes are normal to the curve. An Intersection Snake along the surface-surface intersection is one possibility, but there are many others, for example:

A SubSnake made from an Intersection Snake

Any kind of snake on surface1, that has been used as an edge in the construction of surface2

An Edge Snake of surface1, where surface1 has been constructed from a snake in surface2

(Why not just specify 2 surfaces? Because then an Intersection Snake would have to be constructed internally; and we would have to choose its orientation arbitrarily. A curve or snake gives much more information about the user's intent.)

Each Magnet/surface specifies one of the two surfaces the fillet is to join. Magnets can be used to guide the construction in finding the initial ball position.

Radius = radius desired for fillet

Graph = unitless multiplier of Radius. Default graph * is constant = 1.

The relabel is used to modify the parameter in the v direction on the fillet surface

Examples

BenchFilletsStart.MS2 - Rolling Ball Fillets applied to bench corners. See the SurfaceWorks Forum for components to blend fillets.

BenchFilletsEnd.MS2 - Rolling Ball Fillets blended with the aid of components available from the SurfaceWorks Forum.

Rolling Ball.MS2 – Basic Rolling Ball Fillet with Ruled Surfaces used to finish construction of tangent surfaces.

Rolling Ball Type_6.MS2 – Full Tube Rolling Ball Fillet

Rotated Curve

Parents and characteristic propertiescurve = basis curve

line = axis of rotation
angle = angle of rotation

Description

A Rotated Curve is formed by rotating a basis curve around an axis line through a specified angle, creating a new curve object, which is an exact copy of the basis curve in a new position.

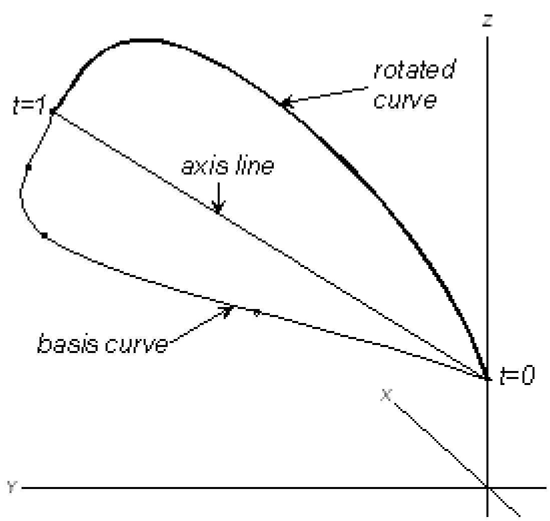
curve can be any curve or snake object. line is a Line object.

angle is specified in degrees. angle = 0 puts the RotatCurve right on top of the basis curve. To determine the direction of increasing angle, use the right hand rule: point your right thumb along the axis line, with the tip toward the t=1 end — your fingers curl (point) in the direction of increasing angle.

relabel is used to relabel the curve. The program's default relabel '*' produces the "natural" labeling, in which the parameter t is distributed the same as for the basis curve.

Example rotatcrv.ms2

'cspline' is the basis curve and 'line_1' is the axis line for the rotated curve 'rotcurve' which has been rotated 60 degrees from the basis curve's position. Check out the right hand rule — point your right thumb along 'line_1', with the tip toward t=1 — angle increases in the direction your fingers point, that is from the basis curve toward the rotated curve (and onward around the axis line).



ROTATCRV.MS2 at Lat 0, Lon 0.

See also: Rotated Point, Rotated Surf

Rotated Point

Parents and characteristic properties

point = basis point
line = axis of rotation
angle = angle of rotation

Description A Rotated Point is created by rotating a basis point around an axis line through a specified angle, creating a new point object.

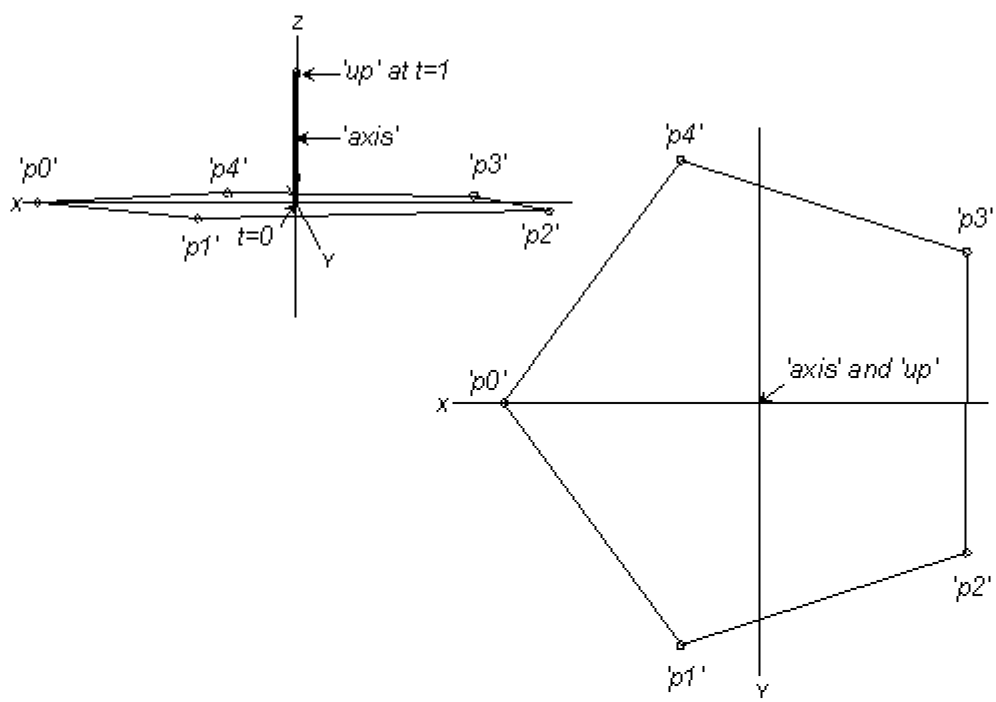
point can be any point object. line is a Line object.

angle is specified in degrees. angle = 0 puts the RotatPoint right on top of the basis point. To determine the direction of increasing angle, you can use the right hand rule: point your right thumb along the axis line, with the tip toward the t=1 end — your fingers curl (point) in the direction of increasing angle.

Rotated entities allow you to build angular relationships and rotational symmetries into your model.

Example Rotated Point.MS2

This model is a regular pentagon. ‘p1’, ‘p2’, ‘p3’, and ‘p4’ are four rotated points all of which use ‘p0’ for the basis point and ‘axis’ for the axis line of rotation (in the right hand figure, ‘axis’ is pointing straight up at you). The angles of rotation are 72, 144, 216, and 288 degrees respectively. Check out the right hand rule — point your right thumb along ‘axis’, from the origin toward ‘up’ — angle increases, as it should, from ‘p0’ to ‘p1’ to ‘p2’, etc. (not from ‘p0’ to ‘p4’ to ‘p3’, etc.).



Rotated Point.MS2 at Lat 2, Lon 90 (left) and Lat 90, Lon 90 (right).

See also: Rotated Curve, Rotated Surface

Rotated Surface

Parents and characteristic properties

surface = basis surface
line = axis of rotation
angle = angle of rotation

A Rotated Surface is formed by rotating a basis surface around an axis line through a specified angle, creating a new surface object, which is an exact copy of the basis surface in a new position.

surface can be any curve or snake object. line is a Line object.

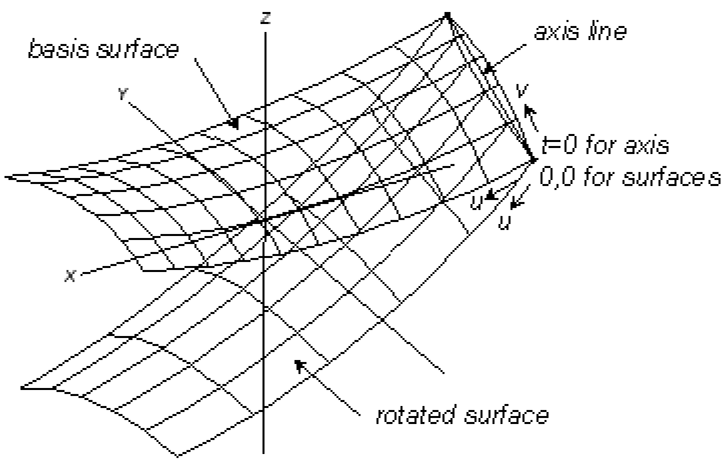
angle is specified in degrees. angle = 0 puts the RotatSurf right on top of the basis surface. To determine the direction of increasing angle, you can use the right hand rule: point your right thumb along the axis line, with the tip toward the t=1 end — your fingers curl (point) in the direction of increasing angle.

For most accurate results, give the OffsetSurf the same divisions as the basis surface.

Example Rotated Surface.MS2

'patch' is the basis surface and 'axis' is the axis line of rotation for the rotated surface 'rotated_surf' which has been rotated 30 degrees from the basis surface's position.

We gave the rotated surface different u-divisions from the basis surface just so it would be easier to differentiate the two surfaces in the black-and-white figure — the surfaces, though, are identical except for their position in space. Note that the u-divisions u-divisions for the two surfaces actually are coordinated: basis surface u-divisions = 10x2, RotatSurf u-divisions = 5x4.



Rotated Surface.MS2 at Lat -30, Lon 60.

See also: RotatCurve, RotatSurf

RPY Frame

Parents and characteristic properties

point where the RPYFrame is located
a parent *frame* (default = '*', the global coordinate system)
3 angles

Description

"RPY" stands for "roll, pitch, yaw", the three standard angles for describing the instantaneous attitude of a ship, and by extension, other vehicles.

An RPYFrame has one additional piece of data: an *orientation* attribute, which is either 0 or 1. This attribute determines the RPYFrame's *handedness* according to the following rules: in a right-handed frame, if you point the index finger of your right hand along the x axis and your middle finger along the y axis, then your thumb points in the positive z direction; in a left-handed frame, substitute your left hand.

Note: To date in SurfaceWorks, frames have always been right-handed coordinate systems. Enough applications have shown up for left-handed frames that we decided to introduce this option with the RPYFrame. If you wish to work exclusively with right-handed frames, just leave the *orientation* attribute at 0 for all your frames.

Construction of an RPYFrame starts by making a parallel copy of the parent *frame* at the location of *point*. Then the three rotations are applied as follows, *in this order*:

1 Roll -- rotation about the x axis of the RPYFrame
2 Pitch -- rotation about the y axis of the RPYFrame
3 Yaw -- rotation about the z axis of the RPYFrame.

The positive direction of any of these rotations is indicated by a right-hand or lefthand rule, depending on the handedness of the parent *frame*: if you grip any one of the axes with the appropriate hand, thumb along the axis in the positive direction, your fingers are curling in the direction of positive rotation.

Finally, if *orientation* is 1, the z axis is reversed. Thus the RPYFrame will have:

opposite “handedness” from its *frame* support when *orientation* is 1,

the same “handedness” as its *frame* support when *orientation* is 0.

Frames made from the Frame3 entity are always right-handed. (Frame3 do not have the *orientation* parameter which could logically allow their handedness to be reversed.)

Sample files: RPY.MS2, RPY-JET.MS2

Ruled Solid

Parents and characteristic properties

surface1 and surface2 = surface parents
relabel = name of Relabel object or ‘*’ for default labeling**Description**

The RuledSolid is a solid between two supporting surfaces. It is generated by taking the point on each of the surfaces at each given u,v and constructing the line (“ruling”) between. w is the parameter along the line.

The supporting surfaces must have similar u,v orientation — the 0,0 corners need to be in a relatively similar place, and u and v must run in similar directions (otherwise the solid could wind up with an unexpected twist).

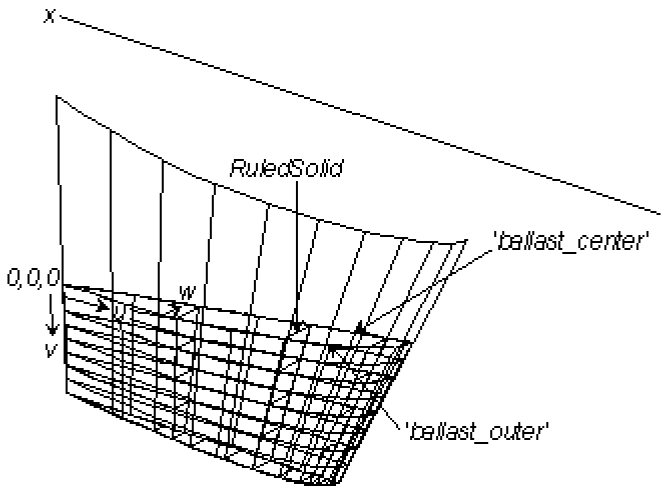
u and v for the solid run in the same directions as u and v for each supporting surface. w is along the rulings. from surface1 to surface2.

relabel is used to control the labeling of the w-direction of the solid. The default relabel ‘*’ produces linear labeling with respect to w.

Example 1 ruledsolidballast1.ms2

This is the outside lead ballast casting for a trapezoidal fin keel. The lead line is adjustable by dragging rings 'r0', 'r1'. You can have the weight schedule open and watch weight and C.G. change as you adjust the lead line.

The lead line at the keel surface is an IntSnake. The SubSurf 's0' below the leadline is projected to centerplane (ProjSurf 's1'), then 's0' and 's1' are the parents for the RuledSolid.



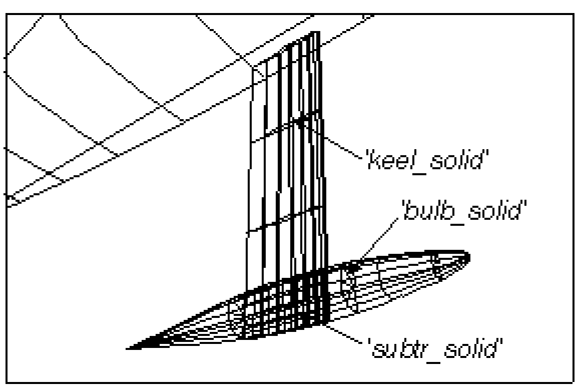
RULEDSOLIDBALLAST1.MS2 at Lat 25, Lon 52.

Example 2 ruledsolidballast2.ms2

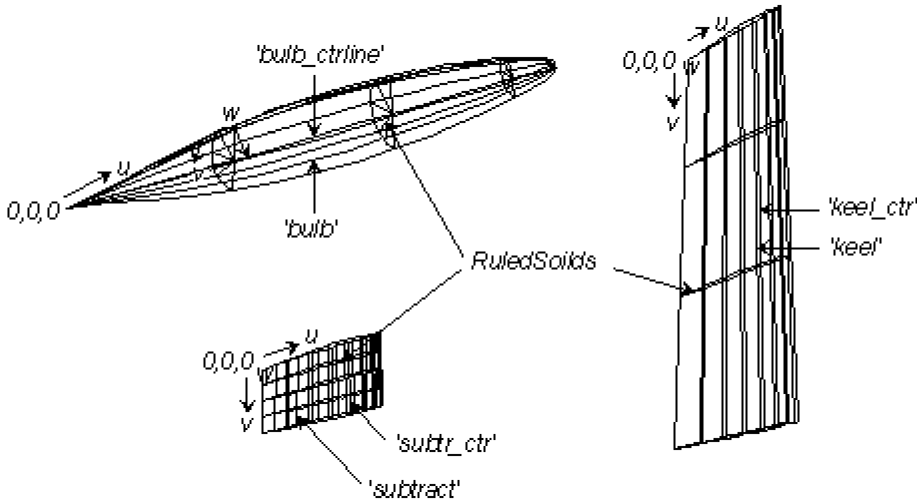
Example 2: ruledsolidballast2.ms2

This is a bulb keel using solids to get weight and C.G. of the fin and bulb combination. ‘bulb_solid’ is a RuledSolid between the bulb surface and a deliberately degenerate RuledSurf that lies along the bulb axis (projection of ‘meridian’ on axis line). ‘keel_solid’ is a RuledSolid between the fin surface and a ProjSurf on the centerplane. Note that ‘keel_solid’ has orientation = 1 to give it a positive volume and weight.

The missing lead where the fin passes through the bulb (we assume here the fin goes all the way through the lead) is accounted for by ‘subtr_solid’. The weight schedule gives the correct weight and C.G. for the combination.



RULEDSOLIDBALLAST2.MS2 at Lat -30, Lon 50.



The three solids in RULEDSOLIDBALLAST2.MS2. The w parameter on the keel and the subtraction solids are transverse lines from 'keel' and 'subtract', respectively, to the centerplane.

See also: B-spline Lofted Solid, Ruled Surface

Ruled Surface

Parents and characteristic properties

Control curves = curve entity or point entity. curve1, curve2

Description A Ruled Surface is created from two curves by connecting points on each parent curve having equal t parameter values with straight lines (the "rulings"). The u parameter for the surface is the same as the t parameter for either curve. curve1 is the v = 0 edge, and curve2 is the v = 1 edge. The surface parameter v is a uniform subdivision of each ruling.

A Ruled Surface interpolates both of its basis curves. Its other two edges are straight lines, viz. the rulings u = 0 and u = 1.

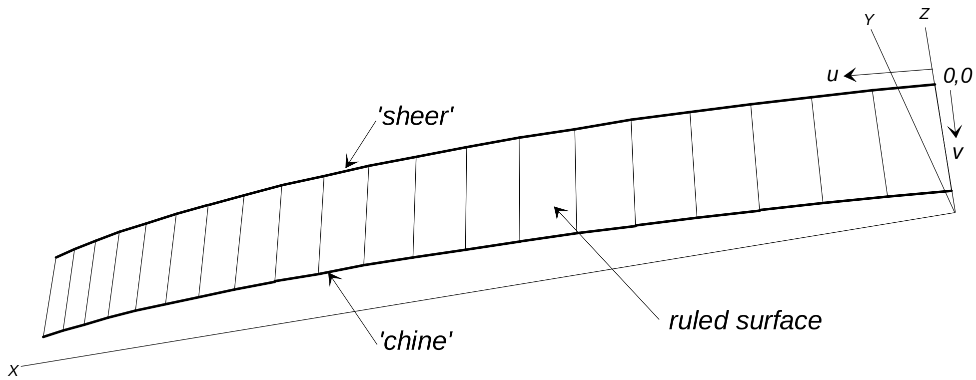
A Ruled Surface is in general not developable. (All developable surfaces are ruled, but NOT all ruled surfaces are developable.) In special circumstances, of course, a Ruled Surface will be developable. For example, if curve2 is a translation copy of 'curve1', the Ruled Surface will be a portion of a general cylinder. If curve1 or curve2 is a single point, the Ruled Surface will be a general cone with this point as apex. Cones and cylinders are always developable.

A B-spline Lofted Surface made with just two control curves is a ruled surface.

If you will be using MSDEV to expand Ruled Surface panel shapes, be sure that the normal orientation is set correctly. MSDEV uses the orientation to determine the direction of thickness for panels — thickness runs in the same direction as the normal orientation. For information on figuring out the direction of a surface's normal, see "Basic Concepts - Parametric Surfaces".

Relabel is used to control the labeling of the v-direction of the surface.

Example: ruled1



ruled1.

‘sheer’ is curve1 and ‘chine’ is curve2 for this ruled surface ‘panel1’. Visibility for the panel turns on only the u-constant lines, which are the rulings.

See also: Developable Surface

Solve Set

Characteristic data

Name -- as with any entity
Layer -- as with any entity
Lock -- as with any entity
Type – 0 (Dormant) or 1 (Active)
Log Tolerance
No. Free – Number of solve operations
Entities – Entities required for solve
User data – as with any entity

The Solve Set allows a fairly general set of constructions that we can't perform with the entity set.

To Solve, we specify a set of entities with the following structure:

One or more free objects, having altogether N degrees of freedom

N pairs of (real, real or point, surface / plane) objectives.

At least one entity in each objective pair has to be a descendant of at least one free object. The Solve Set seeks to adjust each free parameter so as to simultaneously satisfy all of the objectives (zero clearance). Unless the objectives are all linear, which is possible, this is necessarily an iterative procedure. If the iteration converges within the log tolerance, the Solve operation is successful, and all the objects involved are left in updated positions.

The Solve Set, when type is Active, makes the solution a durable part of the model.

Examples

SOLVESET.MS2 – Solves for given v-belt length as well as tangency of the v-belt elements.

Strain Contours

Parents and characteristic properties

First index = value of first index
Last Index = value of last index
s0 = strain on the 0-index contour
Int = strain interval between contours
Surfaces = (Only Expanded Surface and Expanded TriMesh types qualify.)

Description

This entity type provides visualization of strain distribution when surfaces or trimeshes are flattened using Expanded Surface and Expanded TriMesh entity types.

To find the proper values for the characteristic properties, select the Surface for which you want to generate Strain Contours and Tools>Command window (or press 'w'). Type 'strain' and click on OK. A box will popup with the appropriate values to enter. See "Strain" in the Commands chapter for more information.

Examples

StrContours1.ms2 - This starts with Revolution Surface s0, a portion of a sphere. An Expanded Surface s1 is made from it, and Strain Contours x0 (bright red) makes strain contours on s1. The strain range in this case is 0 to -.109.

Also made from s0, an Expanded TriMesh 'I1'. Strain Contours x1 (bright magenta) makes strain contours on I1. These contours are highly similar in location to x0 (take <z> view to compare them).

StrContours2.ms2 - This starts with 8 SubSurfaces ('B' through 'I') made from a Revolution Surface. 8 Expanded Surfaces are made from the SubSurfaces, and Strain Contours 'StrContour1' (bright red) makes strain contours on all the pattern pieces. The strain range in this case is 0 to -.013.

Note: Strain command

With one Expanded Surface or Expanded TriMesh selected. Shows the range of strains required for flattening, and recommended settings for Strain Contours.

SubCurve

Parents and characteristic properties

Degree = blending function B-spline degree: 1, 2, 3, etc.
Control beads = bead. bead1 . . . beadN

Description

A SubCurve is a portion of a parent curve, between bead1 (t=0 on the SubCurve) and beadN (t=1 on the SubCurve). The Control beads designate the parent curve of which the SubCurve is a portion and, therefore, the Control beads must all lie on the same curve.

Most commonly you will probably use only two Control beads, for the two endpoints of the SubCurve. The reason to use more control beads is to change the distribution of the t-parameter along the SubCurve (see discussion below).

Notice that if bead1 is at a larger t-value than bead2 on the parent curve, the parameter t on the SubCurve runs opposite to the way parameter t runs on the parent curve. In fact, one application of a SubCurve is to obtain an identical curve to the parent curve, with the parameter running the opposite way. This is sometimes needed for forming certain surface types (such as a Translation Surface). In this case, bead1 would be at t = 1 and bead2 would be at t = 0 on the parent curve.

Degree is the B-spline degree for the blending functions: 1 = linear, 2 = quadratic, 3 = cubic, etc. As usual, if the number of parents N is less than Degree+1, the splines are automatically demoted to Degree N-1.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, which is the labeling generated by the beads. Relabeling a SubCurve with a Relabel entity will probably compound your relabeling problems, so we recommend you stick to the default for this one!

Internally-defined beads

The internally-defined (or predefined) beads *0 and *1 are available for use as control beads for SubCurve entities, with *0 at the t=0 end of the parent curve and *1 at the t=1 end. But these internally-defined beads do not identify a parent curve. Therefore, at least one of the parent control beads must be a real one, so the curve can be ascertained. Assuming 'e1' and 'e2' are two bead entities on the same curve:

*0 e1
e1 *1
*0 e1 e2 *1
e1 e2 *1

are valid lists of beads, but *0 *1 is not — this last one doesn't identify any particular curve.

Use of internally-defined beads often avoids the need to put a bead at the end of a curve, where there is usually another point already. Internally-defined entities are not visible in the drawing, so you must select them from the Available Entities pane.

Changing the distribution of the t parameter

Sometimes, you may find it useful to redistribute the parameter t along the SubCurve. This will have subtle effects on any surfaces built off the SubCurve, most visibly where the u,v lines for the surface fall.

To change the t-distribution, use more than two beads to define the SubCurve. To see where the t-intervals fall, turn on tickmarks (Visibility = tickmarks) for the SubCurve.

Example 1: Subcurve1

'midsection' (cyan) is the Arc parent curve for 'sidepanel' (red), a SubCurve whose parent beads are 'bead1' and 'bead2', in that order. Thus, the SubCurve begins at 'bead1' (green) and ends at 'bead2' (also green) of the parent curve.

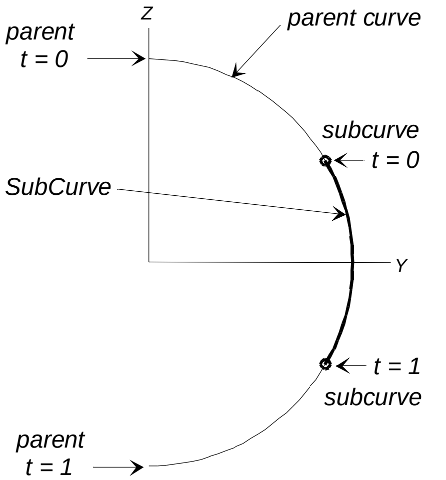


Fig. 1. Subcurve1 (Right View).

Example 2: Polycurve-Subcurve2

This is an example of using a SubCurve to change the distribution of the t-parameter along a curve. This is accomplished by using more than two control beads (in this case, three) to define the SubCurve.

'poly2' (cyan) is the parent curve for the three Beads 'start', 'seatshape', and 'end' (all bright green). 'subcurve' (blue) is a degree-2 SubCurve defined by those beads. Sliding 'seatshape' up and down the length of 'poly2' has quite a dramatic affect on the shape of the surface.

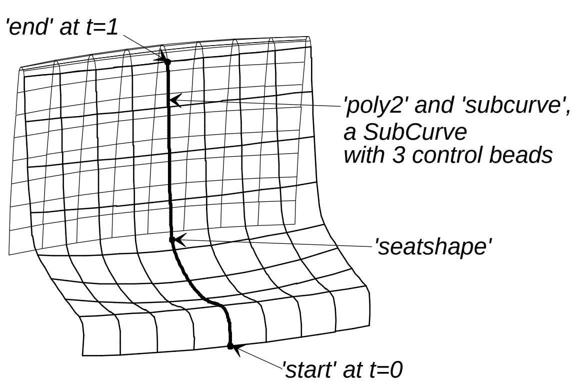


Fig. 2. Polycurve-Subcurve2: initial model.

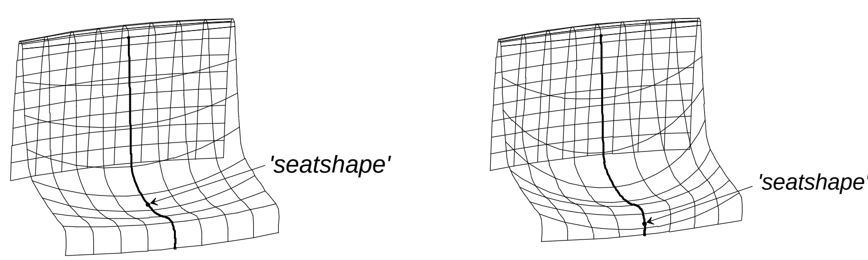


Fig. 3. Polycurve-Subcurve2: with 'seatshape' moved to two different positions.

Experiment with changing the shape: select ‘seatshape’ and slide it to several positions along ‘poly2’ — watch the u,v lines and the surface shape change.

See also: SubSnake, SubSurface, Translation Surface

SubSnake

Parents and characteristic properties

Degree = blending function B-spline degree: 1, 2, 3, etc.
Control rings = ring. ring1 . . . ringN

Description

A SubSnake is a portion of a parent snake, between bead/ring1 (t=0 on the SubSnake) and bead/ringN (t=1 on the SubSnake). The Control beads designate the parent snake of which the SubSnake is a portion and, therefore, the Control beads must all lie on the same snake.

Notice that if bead/ring1 is at a larger t-value than bead/ring2 on the parent snake, the parameter t on the SubSnake runs opposite to the way parameter t runs on the parent snake.

Degree is the B-spline degree for the blending functions: 1 = linear, 2 = quadratic, 3 = cubic, etc. As usual, if the number of parents N is less than Degree+1, the splines are automatically demoted to Degree N-1.

Relabel is used to relabel the snake. The program's default Relabel ‘*’ produces the "natural" labeling, which is the labeling generated by the beads / rings. Relabeling a SubSnake with a Relabel entity will probably compound your relabeling problems, so we recommend you stick to the default for this one!

Internally-defined rings

The internally-defined (or predefined) rings *0 and *1 are available for use as Bead/Ring parents for SubSnake entities, with *0 at the t=0 end of the parent snake and *1 at the t=1 end. But these internally-defined rings do not identify a parent snake. Therefore, at least one of the parent control points must be a real one, so the snake can be ascertained.

Assuming ‘r1’ and ‘r2’ are two ring entities on the same curve:

*0 r1
r1 *1
*0 r1 r2 *1
r1 r2 *1

are valid lists of rings, but *0 *1 is not — this last one doesn’t identify any particular snake.

Use of internally-defined beads / rings often avoids putting a bead or ring at the end of a snake, where there usually already is another point. Internally-defined entities are not visible in the drawing, so you must select them from the Available Entities pane.

Changing the distribution of the t parameter

Sometimes, you may find it useful to redistribute the parameter t along the SubSnake. This will have subtle effects on any surfaces built off the SubSnake, most visibly where the u,v lines for the surface fall.

To change the t-distribution, use more than two bead / rings to define the SubSnake. To see where the t-intervals fall, turn on tickmarks (Visibility = tickmarks) for the SubSnake.

Example 1: Subsnake1

'end1' and 'end2' (both green) are Rings lying on the parent B-spline Snake 'bsnake' (magenta) at s = .3 and s = .7 respectively. 'segment' (blue) is the portion of 'bsnake' that lies between the two Rings. 'end1' is the t = 0 end of 'segment' and 'end2' is the t = 1 end.

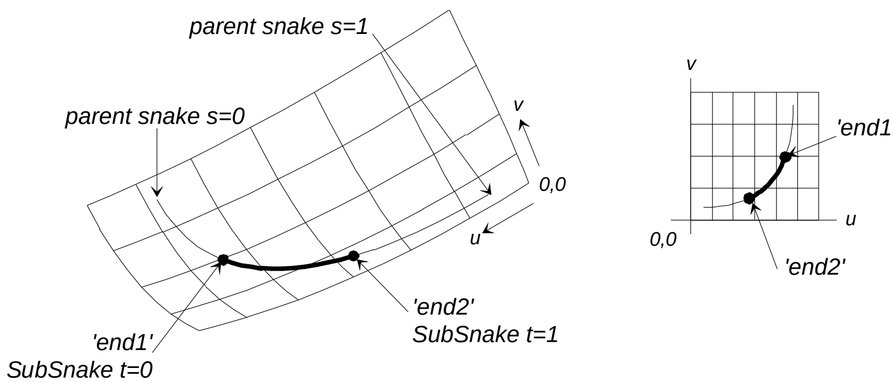


Fig. 1. Subsnake1

Example 2: Subsnake2

In this case, 'bsnake' (magenta) is the same parent B-spline Snake as 'bsnake' in the previous example, but the Rings 'end1' and 'end2' are in different positions: 'end1' is at the parent snake's t = 1 and 'end2' is at the parent snake's t = 0. Since the t-value for the first ring is larger than that of the second ring, the parameter t on the SubSnake 't_reversed' (blue) runs in the opposite direction to the parameter t on the parent snake.

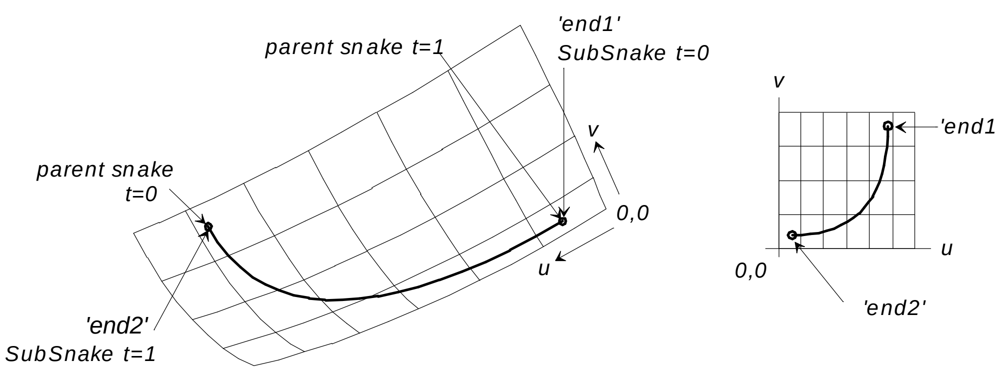


Fig. 2. Subsnake2

See also: SubCurve, SubSurface, Trimmed Surface

SubSurface

Parents and characteristic properties

Bounding snakes = snake, magnet or ring. two or four required

Description

A SubSurface is a parameterized portion of a surface, bounded by snakes. You specify a SubSurface by naming either two or four Bounding snakes, all of which must belong to the same surface.

Two snakes: snake1, snake2. Corresponding points on snake1 and snake2 are connected with straight lines in the u-v parameter space of the surface. The u=0, v=0 corner of the SubSurface is at the t=0 end of snake1. snake1 forms the v=0 edge of the SubSurface; snake2 forms the v=1 edge — thus snake1 and snake2 form opposite edges. The u parameter is the same as the t parameter of snake1.

Either snake can be a magnet or a ring; this allows creation of three-sided SubSurfaces.

Four snakes: snake1, snake2, snake3, snake4. The four snakes nominally connect at their endpoints. The order of the four snakes is snake1, snake2, snake3, snake4; and snake4 connects back to snake1. The snakes should be picked in sequence, going around the loop in either a clockwise or counterclockwise direction. Picking the snakes out of order will give unpredictable results.

The u=0, v=0 corner of the surface is where snake1 and snake4 join. snake1 forms the v=0 edge of the SubSurface; snake2 forms the u=1 edge — thus snake1 and snake2 form adjacent edges. The u parameter is the same as the t parameter of snake1 (or 1 - t, if snake1 is used in reverse).

One (or more) of the snakes can be a magnet or a ring, allowing creation of three-sided SubSurfaces.

Troubleshooting tip: If the SubSurface doesn't fill the boundary as expected, or turns out looking like a spider web, the most likely problem is that the bounding snakes are not in sequential order. To fix this, Edit Definition and either pick the snakes again or use the Selection Set pane ↑ and / or ↓ buttons to reorder the list.

A second common reason for a SubSurface not filling its boundary is the bounding snakes not forming a closed loop in the u-v parameter space. (This is not easy to see in 3D if the boundary includes all or part of a degenerate edge of the surface.)

Comparison with trimmed surfaces

The SubSurface, as a portion of some other surface patch, has some similarities to the Trimmed Surface and to "trimmed surface" entities available in other advanced surface modeling programs, but also some differences. The key difference is that a standard trimmed surface is not a parametric surface in and of itself (it has no mesh of u,v parameter lines on it, and none of the corresponding internal structure); it is simply a portion of a parametric surface, used for drawing and viewing. The SubSurface is a complete parametric surface. Both Trimmed Surfaces and

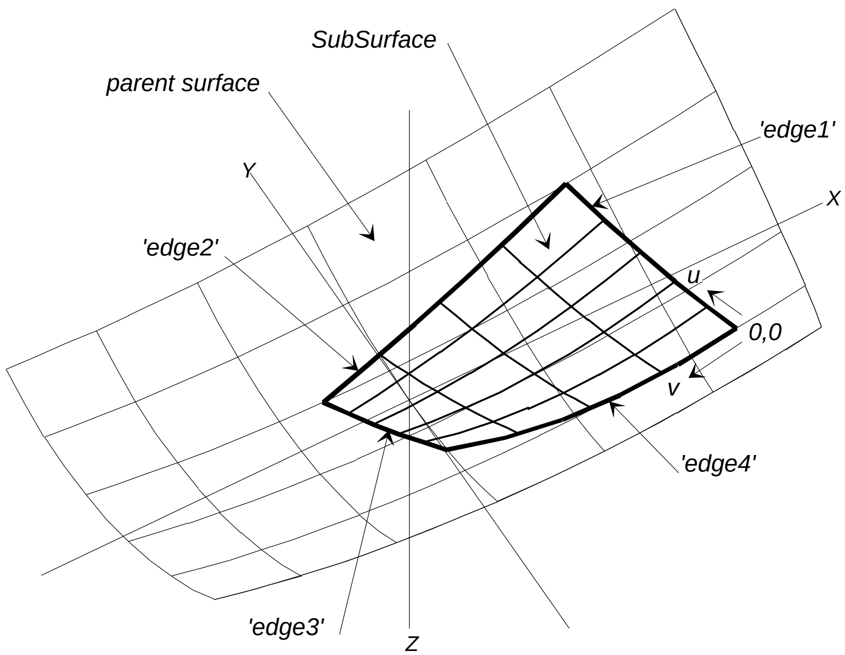
SubSurfaces can be used as parents for magnets and snakes, but only SubSurfaces can be used for Offset or Mirrored Surfaces.

Another major difference is that trimmed surface boundaries can be arbitrarily complex and can include one or more holes. SubSurface boundaries are limited to two or four snakes and cannot include holes.

While in some programs a trimmed surface can be composed of two or more disjoint parts, disjoint SubSurfaces and Trimmed Surfaces are not supported in SurfaceWorks. In such a case, you would need to create a SubSurface or a Trimmed Surface for each disjoint region.

Example 1: Subsurf1

In this example, all the magnets (‘M1’ - ‘M5’) and therefore all the snakes they define (‘edge1’ - ‘edge4’) lie in a surface called ‘patch’ (dark green). ‘subsurf’ (blue) is a SubSurface of this patch, bounded by the four snakes ‘edge1’ (cyan), ‘edge2’ (green), ‘edge3’ (red), and ‘edge4’ (yellow).

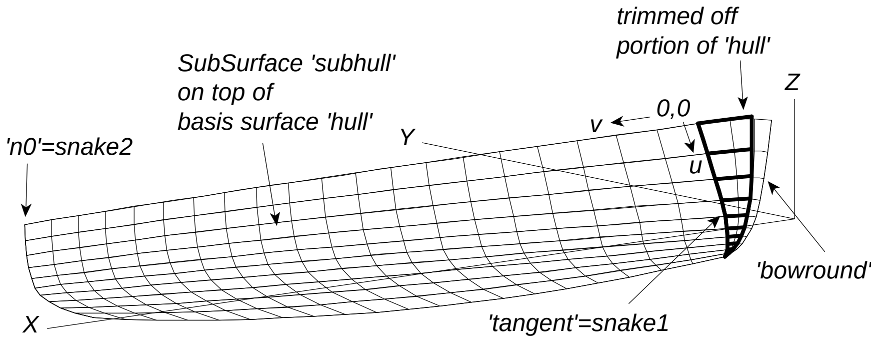


Subsurf1.

Example 2: Subsurf2

SubSurface ‘subhull’ (white), based on the C-spline Lofted Surface ‘hull’ (green) neatly “trims off” the portion of ‘hull’ forward of the B-spline Snake ‘tangent’ (yellow). Together, ‘subhull’ and the bow rounding surface (‘bowround’; blue) form a smooth and durable representation of the bow-rounded hull. To see this more clearly, select ‘hull’ and ‘tangent’ and hide them.

The SubSurface ‘subhull’ is defined between just 2 snakes: ‘tangent’, a B-spline Snake on ‘hull’ (it is also a master curve for the bow rounding B-spline Lofted Surface), and ‘n0’, an Edge Snake along the aft end of ‘hull’. Both snakes have the same t-orientation — from the sheerline downwards.



Subsurf2.

See also: SubCurve, SubSnake, Trimmed Surface

Surface TriMesh

Parents and characteristic properties

degree = degree of subdivision
surface = Surface, Trimmed Surface or Composite Surface
nu = no. of triangles in u-direction
nv = no. of triangles in v-direction
style = 0 – Forward   1 – Reverse   2 - Alternating

Description

The SurfaceTriMesh entity type, parented by a parametric or trimmed surface, or Composite Surface. This creates a TriMesh from a surface. It has 3 integer parameters: nu, nv, style. These parameters have no effect when the parent is a TrimSurf or CompSurf -- in those cases, you just get the object's triangulation. When the parent is a parametric surface, it is divided first into rectangles (in u,v parameter space) along u=constant and v=constant parameter lines, then each rectangle is split into two triangles.

nu controls the number of triangles in the u direction, with these options:

nu < 0 -- uniform subdivision into -nu strips, each covering 1/(-nu) of the u parameter space.

nu = 0 -- similar uniform subdivision as nu<0, but taking nu from the basis surface u-divisions x subdivisions

nu > 0 -- the surface is first divided at any degree-1 breaklines, then the intervals between breaklines are uniformly subdivided into a total of nu strips

nv has similar meaning for the v-direction subdivision

style selects forward diagonals, reverse diagonals, or alternating diagonals for the division of each rectangle into two triangles.

Example

SurfTriMesh1.ms2

‘patch’ (the basis surface) is a Translation Surface. Surface TriMesh 'I0' has nu = 8, nv = 4, and style = 0. It therefore divides 'patch' into 8 x 4 x 2 = 64 triangles. These are uniformly distributed in both u and v directions, because 'patch' has no degree-1 breaklines.

See Also: Light TriMesh, Poly Trimesh, Copy Trimesh, Trimesh B-spline Snake, Trimesh Edge Snake, Trimesh Intersection Snake, Trimesh Projected Snake, Trimesh SubSnake, Trimesh Magnet, Trimesh Projected Magnet, Trimesh Ring

Sweep Surface

Parents and characteristic properties

Bead/Ring = bead or ring. designates the “Path” curve
Shape = curve entity
Guide = curve entity

Description

A Sweep Surface is created by sliding or “sweeping” a copy of a Shape curve along a “Path” curve, designated by Bead/Ring. During this process, the Shape curve copy maintains a constant orientation with respect to the local tangent to the “Path.” Consequently, the Sweep Surface has the character of a bent extrusion — Shape providing the extrusion cross section, and “Path” specifying the bending (or no bending, if “Path” happens to be straight).

What then is the role of the Guide curve? Notice that the description in the preceding paragraph is incomplete in one respect: the tangent to “Path” furnishes one direction for orienting the copies of Shape, but leaves one degree of freedom — rotation about the tangent — unspecified. The Guide curve controls this degree of freedom.

You might think of the generation of a Sweep Surface in terms of a moving “frame,” or local 3D coordinate system, attached to “Path.” The first leg (x-axis) of the frame is a unit vector along the positive tangent to “Path,” at parameter value u. The second leg (y-axis) of the frame is perpendicular to the first and is rotated so as to aim as nearly as possible toward the point taken from Guide at parameter value u. The third leg (z-axis) of the frame is perpendicular to the first two legs. Each copy of Shape has the same position in the moving frame.

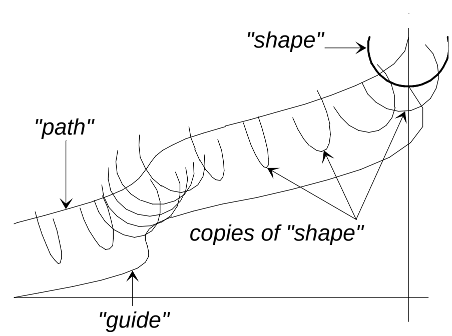


Fig. 1. Sweep Surface shape, path, and guide.

Bead/Ring actually serves two functions:

1 it designates which curve or snake to use as the “Path”

2 it specifies a position along the “Path” for the base frame into which Shape is copied

The u-direction is along “Path”; u is equal to “Path’s” parameter t. The v-direction is along Shape; v is equal to Shape’s parameter t.

Example 1: Sweep1

This Sweep Surface example is an oval extrusion bent into a symmetric U. ‘path’ (blue), designated by ‘bead1’ (bright green), is a B-spline Curve giving the form of the U bend. ‘shape’ (cyan) is a B-spline Curve giving the cross-section. ‘shape’ is positioned in relationship to ‘bead1’, at the middle of ‘path’. The base frame into which ‘shape’ is copied is located at ‘bead1’, with x-axis tangent to ‘path’ and y-axis vertical. ‘guide’ (brown) is a Relative Curve with the same vertical offset at each end of ‘path’, ensuring that each copy of ‘shape’ has a vertical orientation. The ‘shape’ copies are exactly like ‘shape’.

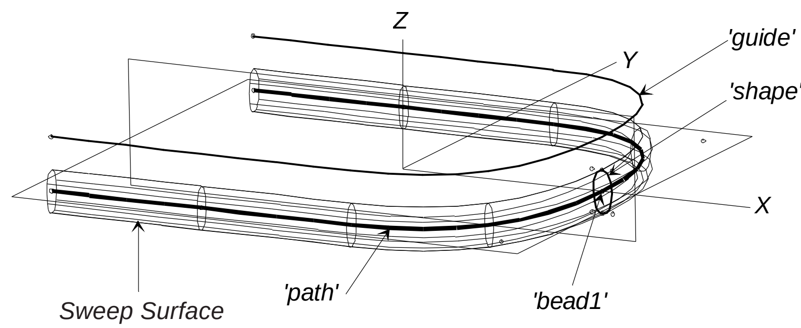


Fig. 2. Sweep1

Example 2: Sweep3

This model is a chute for a waterslide (Fig. 3). ‘path’ is a freeform B-spline Curve; ‘shape’ is an Arc of about 200 degrees (Fig. 4, left). ‘guide’ is also a B-spline Curve, with its control points below and outboard of the corresponding control points on ‘path’ (Fig. 4, middle and right), so as to bank the turns.

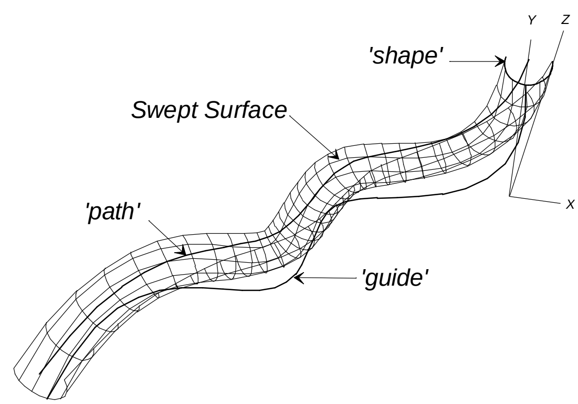


Fig. 3 Sweep3

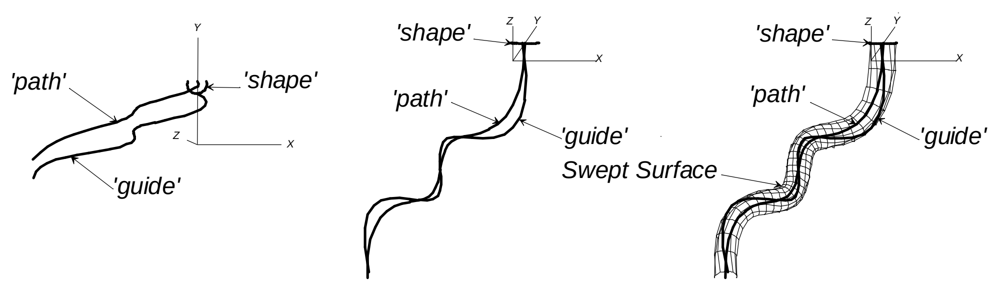


Fig. 4. Sweep3.

Example 3: Helix-Sweep4

This example is the same one as used for the Helix entity description. The hose surface is a Sweep Surface using the following parents:

- the Bead ‘bead1’ (green; at the t=0 end of ‘path’); designates the Helix as the path curve for the surface
- the Arc ‘shape’ (magenta; to see it, zoom in on the area around ‘bead1’)
- the Relative Curve ‘guide’ (blue)

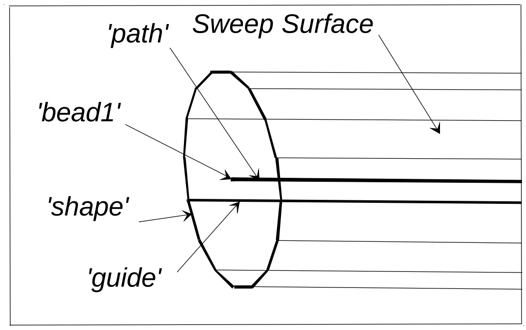


Fig. 5. Helix-Sweep4: lat = 17, lon=7, tilt= -97

See also: Translation Surface

Tabulated Curve

Parents and characteristic properties

filename = filename of .3DA or .PAT file
polyline number = number of polyline in file
frame/point = name of a frame or point object or '*' for the default

Description filename is the filename of a .3DA (or .PAT) file. If no extension is given, SurfaceWorks assumes .3DA.

A .3DA file (see sample contents and notes in example below) consists of a series of records (file lines) each having four numbers: a pen number and X, Y, Z coordinates. A zero pen means "move to this point without drawing". Therefore, if we let nlinks equal the number of divisions in a polyline, a polyline with nlinks divisions is represented by one pen = 0 record followed by nlinks records with nonzero pen. The product of t-divisions × t-subdivisions must be correctly set equal to nlinks, or a geometry error occurs. Each 0-pen record is counted as the start of a new polyline.

polyline no. is the number of the polyline to be taken from the .3DA file: 1 for the first, 2 for the second, etc.

frame/point is the name of a frame or point object or '*' for the default:

When frame/point is a point, it specifies an insert origin, and point's coordinates are added to the polyline coordinates. This results in a shift (but no rotation).

When frame/point is a frame, the file coordinates are interpreted as frame coordinates x,y,z. This generally results in a shift plus a rotation.

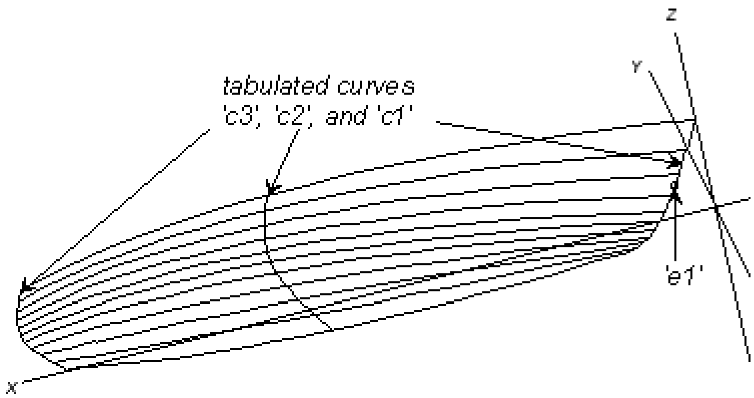
When frame/point is the default '*', there is no shift or rotation.

relabel is used to relabel the curve. The program's default relabel '*' produces a labeling which corresponds to the labeling of the original polyline points; i.e., the first point is at t = 0; the second is at t = 1/nlinks, the third is at 2/nlinks.... and the last is at t = 1.

Note: A TabCurve has less inherent smoothness than most native SurfaceWorks curves. It is actually a polyline, like a type-1 BCurve. You cannot subdivide a TabCurve to produce a smoother version of the same curve.

Example tabcurve.ms2 (needs demomcs.3da)

This example produces a model identical to the DEMO model. However, in this case, instead of generating the master curves itself, the model reads them in from a file DEMOMCS.3DA (see contents below). Therefore, the shape of the hull depends entirely on the contents of that file.



TABCURVE.MS2 at Lat -30, Lon 60.

The three TabCurves are named ‘c1’, ‘c2’, and ‘c3’. For each of them, filename is DEMOMCS, and frame/point is the default ‘*’ which produces no shift or rotation of the curves. polyline no. for ‘c1’ is 1, for ‘c2’ is 2, and for ‘c3’ is 3.

The bead ‘e1’ is included to show that a TabCurve, like any other curve, will support beads and other objects depending on curves.

Contents of DEMOMCS.3DA

0	0.000	0.000	3.600	pen=0 point (1st point) on 1st polyline ‘c1’
11	0.511	0.000	2.493	1st of 10 records with nonzero pen for 1st polyline; nlinks for ‘c1’ = 10
11	0.951	0.000	1.569	
11	1.321	0.000	0.829	
11	1.619	0.000	0.273	
11	1.845	0.000	-0.099	
11	2.045	0.000	-0.355	
11	2.260	0.000	-0.564	
11	2.491	0.000	-0.724	
11	2.738	0.000	-0.836	
11	3.000	0.000	-0.900	
0	15.000	4.815	2.560	pen=0 point (1st point) on 2nd polyline ‘c2’
11	15.000	4.869	1.835	
11	15.000	4.847	1.204	
11	15.000	4.749	0.667	
11	15.000	4.575	0.226	
11	15.000	4.325	-0.121	
11	15.000	3.921	-0.403	
11	15.000	3.286	-0.649	
11	15.000	2.421	-0.860	
11	15.000	1.326	-1.035	
11	15.000	0.000	-1.175	
0	30.000	3.500	2.760	pen=0 point (1st point) on 3rd polyline ‘c3’
11	30.000	3.480	2.218	
11	30.000	3.420	1.742	

```
11   30.000    3.320    1.334
11   30.000    3.180    0.994
11   30.000    3.000    0.720
11   30.000    2.720    0.504
11   30.000    2.280    0.336
11   30.000    1.680    0.216
11   30.000    0.920    0.144
11   30.000    0.000    0.120
```

See also: Tabulated Point, Tabulated Surface, WireFrame

Tabulated Point

Parents and characteristic properties

filename = filename of .3DA or .PAT file
point number = number of points in file
frame/point = name of a frame or point object or '*' for the default

Description

filename is the filename of a .3DA (or .PAT) file. If no extension is given, SurfaceWorks assumes .3DA.

A .3DA file (see sample contents in example below) consists of a series of records (file lines) each having four numbers: a pen number and X, Y, Z coordinates. TabPoint ignores the pen value, and just counts each record as an individual point. The file must have at least as many points in it as the point no. specified, or a geometry error occurs.

point no. is the sequence number of the point to be taken from the .3DA file: 1 for the first, 2 for the second, etc.

frame/point is the name of a frame or point object or '*' for the default:

When frame/point is a <u>point</u>, it specifies an insert origin, and point's coordinates are added to the TabPoint coordinates. This results in a shift (but no rotation).

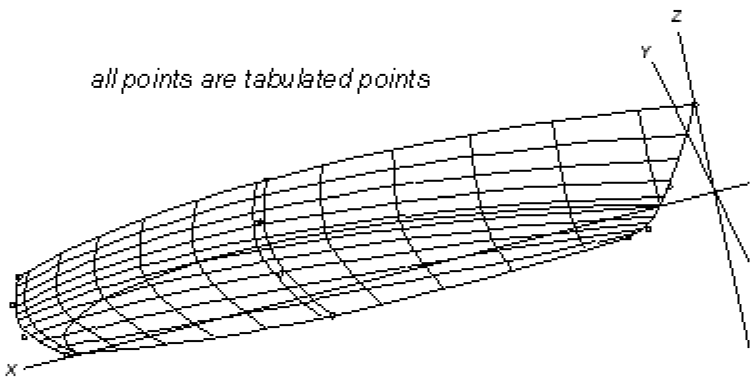
When frame/point is a <u>frame</u>, the file coordinates are interpreted as frame coordinates x,y,z. This generally results in a shift plus a rotation.

When frame/point is <u>the default</u> '*', there is no shift or rotation.

Example tabpoint.ms2 (needs demopts.3da)

The model generated by this file is identical to the DEMO model. If you use Edit / Model File, you can see the difference is that where the DEMO.MS2 has all of its control points "hard-wired," TABPOINT.MS2 gets its control points by reading the file DEMOPTS.3DA (see contents below). Thus, the shape of the TABPOINT model can be controlled in many ways by changing the contents of DEMOPTS.3DA.

The 4 TabPoints defining the bow master curve are named 'P11' - 'P14' (from top to bottom), those for the middle master curve 'P21' - 'P24' (top to bottom), those for the stern master curve 'P31' - 'P34' (also top to bottom). For all the TabPoints, filename is DEMOPTS, and frame/point is the default '*' which produces no shift or rotation of the points. point no. for 'P11' is 1, for 'P12' is 2, for 'P13' is 3, for 'P14' is 4, for 'P21' is 5, for 'P22' is 6, etc.



TABPOINT.MS2

Contents of DEMOPTS.3DA

```
0   0.000    0.000    3.600
1   1.367    0.000    0.602
1   2.324    0.000   -0.800
1   3.000    0.000   -0.900
0  15.000    4.815    2.560
1  15.000    5.046    0.628
1  15.000    3.603   -0.870
1  15.000    0.000   -1.175
0  30.000    3.500    2.760
1  30.000    3.500    1.320
1  30.000    2.500    0.120
1  30.000    0.000    0.120
```

See also: Tabulated Curve, Tabulated Surface, WireFrame

Tabulated Surface

Parents and characteristic properties

filename = filename of .PAT file
patch number = number of patch in file
frame/point = name of a frame or point object or '*' for the default

Description

filename is the filename of a .PAT file. If no extension is given, SurfaceWorks assumes .PAT.

A .PAT file (see sample contents and notes in example below) is a special case of a .3DA file, in which one or more rectangular meshes (patches) is represented by a set of nv + 1 polylines, each having nu + 1 points. Individual patches are separated by an extra record having zero pen. A polyline of nu + 1 points consists of one pen = 0 record followed by nu records with nonzero pen. u-divisions × u-subdivisions is required to be equal to nu, and v-divisions × v-subdivisions is required to be equal to nv, or a geometry error occurs.

patch no. is the number of the patch to be taken from the .PAT file: 1 for the first, 2 for the second, etc.

frame/point is the name of a frame or point object or '*' for the default:

When frame/point is a point, it specifies an insert origin, and point's coordinates are added to the surface coordinates. This results in a shift (but no rotation).

When frame/point is a frame, the file coordinates are interpreted as frame coordinates x,y,z. This generally results in a shift plus a rotation.

When frame/point is the default '*', there is no shift or rotation.

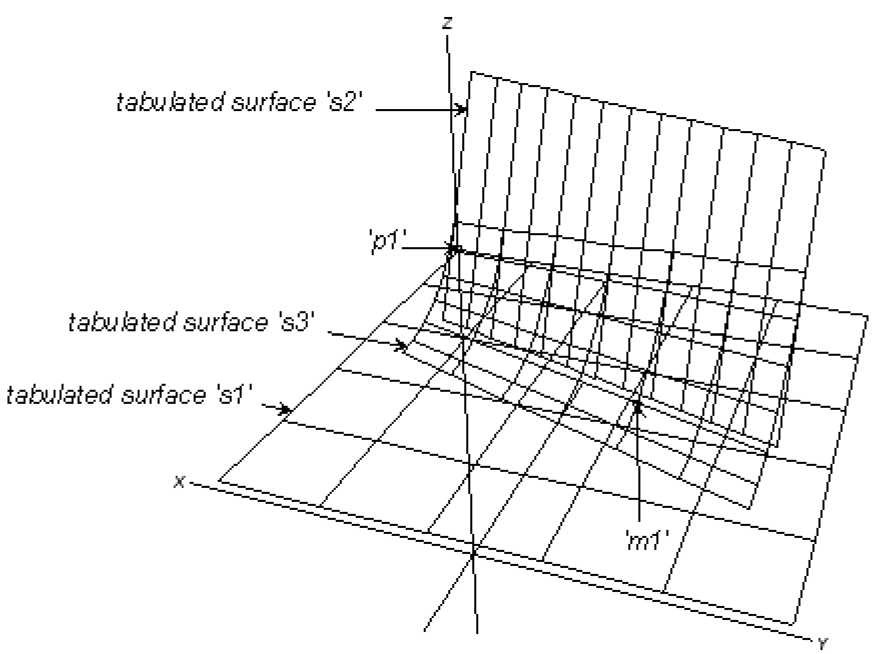
The u-direction of the TabSurf is in the direction of the polylines.

Note: A TabSurf has less inherent smoothness than most native SurfaceWorks surfaces. A TabSurf is a faceted model of the original surface and cannot be further subdivided inside SurfaceWorks.

Example tabsurf.ms2 (needs fillet.pat)

FILLET.PAT is a patch file (see partial contents below) made from a model having three surfaces: two plane patches, with a fillet surface between them. The TabSurf 's1' is the first of these patches, 's2' is the second, and 's3' is the third. The three patches use for insert location (frame/point) the Point 'p1' (at X=0, Y=0, Z=1), which has the effect of adding 1.0000 to the Z-coordinate of each point read from the file. Since frame/point is a point, there is no rotation of the patches, just a 1-unit shift in Z. Moving 'p1' would shift the insert location of the entire set of TabSurfs, again without rotating them.

The magnet 'm1' is added to show that TabSurfs, like other surfaces, can parent magnets.



TABSURF.MS2 at Lat 30, Lon 160.

Partial contents of fillet.pat

0	0.000	0.000	0.000	pen=0 point (1st point) on 1st polyline of 1st patch ('s1')

10	0.200	0.000	0.000	
10	0.400	0.000	0.000	
10	0.600	0.000	0.000	
10	0.800	0.000	0.000	
10	1.000	0.000	0.000	6th (last) point of 1st polyline; u-divisions x u-subdivisions for 's1' = 5; nu + 1 = 5 + 1 = 6
0	0.000	0.200	0.000	pen=0 point (1st point) on 2nd polyline
10	0.200	0.200	0.000	
10	0.400	0.200	0.000	
10	0.600	0.200	0.000	
10	0.800	0.200	0.000	
10	1.000	0.200	0.000	
0	0.000	0.400	0.000	pen=0 point (1st point) on 3rd polyline
10	0.200	0.400	0.000	
10	0.400	0.400	0.000	
10	0.600	0.400	0.000	
10	0.800	0.400	0.000	
10	1.000	0.400	0.000	
0	0.000	0.600	0.000	pen=0 point (1st point) on 4th polyline
10	0.200	0.600	0.000	
10	0.400	0.600	0.000	
10	0.600	0.600	0.000	
10	0.800	0.600	0.000	
10	1.000	0.600	0.000	
0	0.000	0.800	0.000	pen=0 point (1st point) on 5th polyline
10	0.200	0.800	0.000	
10	0.400	0.800	0.000	
10	0.600	0.800	0.000	
10	0.800	0.800	0.000	
10	1.000	0.800	0.000	
0	0.000	1.000	0.000	pen=0 point (1st point) on 6th (and last) polyline; v-divisions x v-subdivisions for 's1' = 6; nv + 1 = 5 + 1 = 6
10	0.200	1.000	0.000	
10	0.400	1.000	0.000	
10	0.600	1.000	0.000	
10	0.800	1.000	0.000	
10	1.000	1.000	0.000	
0	0.000	0.000	0.000	extra pen=0 record = end of patch
0	0.342	0.121	0.000	pen=0 (first) point of 1st polyline on 2nd patch ('s2')
11	0.360	0.186	0.000	
...				

See also: Tabulated Point, Tabulated Curve, WireFrame

Tangent Boundary Surface

Note to SurfaceWorks **1.0 users**:The version 2.0 Tangent Boundary Surface is a new, more comprehensive Tangent Boundary Surface entity, which includes G2 continuity and multiple controls for the interior of the surface. If you have Tangent Boundary Surfaces in your 1.0 models, they will be converted to the new type Tangent Boundary Surface. For the obsolete entity description, please refer to your version 1.0 User's Guide.

Parents and characteristic properties

Number of u and v control values = number of surface controls in the u and v directions
Bounding curves = curve entity or point entity. four bounding curves
G0 | G1 | G2 = edge tangency curvature continuity
Control values = offset of surface control from basis surface

Description

A Tangent Boundary Surface is defined by four boundary curves (and / or snakes and / or points), a set of control values that influence the interior shape of the surface, and a set of edge conditions. The four bounding curves define a basis boundary surface which can be further shaped using the control values and by imposing edge continuity conditions. Tangency and curvature continuity conditions can be imposed along any combination of edges that are defined by snakes, non-linear planar curves, or points.

Bounding curves

Bounding curves for a Tangent Boundary Surface can be any combination of four curve and / or snake entities; but if you want to impose tangency/curvature conditions on the edges they define, the curves must be either snakes or planar curves, and they cannot be lines. In addition, you can use a point, bead, magnet, or ring for one (or more) of the four curves. Use of a point entity for one edge forms a 3-sided patch that spans the three remaining edges. (Use of multiple points for adjacent edges may give unanticipated results.)

The four bounding curves nominally connect at their endpoints. The order of the four curves is curve1, curve2, curve3, curve4; and curve4 connects back to curve1. Pick the curves (including any point used for an edge) in sequence, going around the loop in either a clockwise or counterclockwise direction.

Note: If you want the surface and its control sliders to be displayed in approximately the same visual orientation as the spreadsheet control value cells, e.g. so that the top-left slider corresponds to the top-left control value, etc., making it easy to compare the two, here are a couple of tips:

When choosing the bounding curves, put your model in a view that (approximately) shows one bounding curve at the "bottom", one on "each side", and one at the "top", then choose the curves beginning with the bottom one and going counterclockwise around the boundary. See also "Color coding of control values:" on page 418.

or

If the surface already exists, look at the surface's entity orientation marks (if they're not on, you can turn them on using Tools>Options>General>Auto Orientation>Single Selected Entity or View>Entity Orientation), then rotate the model until the v-arrowhead points upward and the u-arrowhead points to the right.

The Tangent Boundary Surface meets all four boundary curves, if these curves actually join end-to-end in sequence to form a closed loop. The four edges are identified as follows:

curve1 v = 0
curve2 u = 1
curve3 v = 1
curve4 u = 0

The u=0, v=0 corner of the surface is where curve1 and curve4 join. The u parameter is the same as the t parameter of curve1 (or 1 - t, if curve1 is used in reverse).

It is assumed that the boundary curves meet each other at the corners. However, this condition is neither enforced nor checked. If one or more corners are "open," you still get a continuous surface patch, but it no longer meets all the edges.

Troubleshooting tip:If the surface doesn't fill the boundary as expected, or turns out looking like a spider web, the most likely problem is that the bounding curves are not in sequential order. To fix this, Edit>Definition and either pick the curves again or use the Selection Set pane ⬆ and/or ⬇ buttons to reorder the list.

Changing bounding curves:

When you change one or more bounding curves (that is, use a different curve as the parent), continuity resets to G0 for those curves that change their position in the list. The control values for the surface remain unchanged.

Control values

The interior of the Tangent Boundary Surface is shaped by a set of control values with a user-specified number of values in each of the surface's u and v directions. These control values are used in the calculation of B-spline functions that are used to define the surface's shape. The larger you make the Number of u and v control values, the more control you have on the shape of the surface. The maximum total number of control values is 100.

A control value is an offset from an (imaginary) point on the basis boundary surface in a direction normal to that surface. The control values are displayed in two ways:

- As point-on-a-line sliders in the graphic image, displayed when the Insert/Edit dialog is open (Fig. 1). When you are first creating a Tangent Boundary Surface, you must specify the bounding curves and click Apply before the set of control sliders will appear in the graphic display. (If additionally you set the number of control values and the edge continuities you want before clicking Apply, you'll immediately be ready to shape your surface.) After that, the slider controls will be displayed whenever you have the Edit dialog open for the surface and the cursor focus is not in the "Bounding curves /continuity" field.

- As numerical values in the spreadsheet on the Control Values tab of the Insert/Edit dialog.

The graphic display and the Control Values tab are color coded and synchronized so changes made in one will update in the other (click Apply or OK to make dialog changes update in the graphic display).

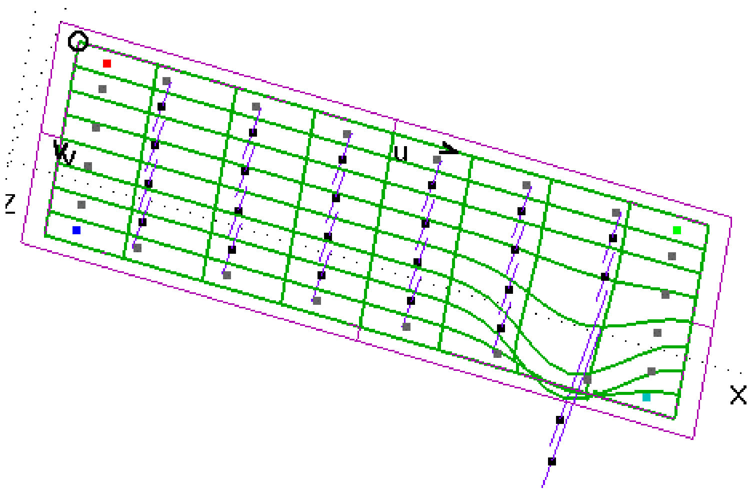


Fig. 1. Basis boundary surface with 2 control sliders dragged downward.

Color coding of control values:

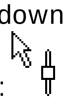
The slider / value in each corner of the set of control values is color coded, so you easily can see which slider control in the graphic display corresponds to which value in the table on the Control Values tab. For example, the value in the spreadsheet and the slider point nearest the surface's u=0, v=0 corner are displayed in red.

Grayed values in the table and point controls that have no normal line to slide along are fixed at zero by your choice of continuity:

- For G1 continuity, the outer row of values corresponding to the curve will be fixed at zero.
- For G2 continuity, the outer two rows of values corresponding to the curve will be fixed at zero.

To set or change control values:

You can specify control values by sliding the controls in the graphic display or by entering values on the Control Values tab:

- To set a control value using a slider in the graphic display, drag the point of the control up or down along the normal line. The cursor icon for the slider points looks like this:  .
- To set a control value via the spreadsheet on the Control Values tab, simply type in the new value; click Apply (or OK) to update the graphical image. In the spreadsheet, Tab and Shift+Tab move horizontally to the next and previous cells, respectively; the up and down arrow keys move to the cells above and below, respectively. You can select one or more cells, copy (Ctrl+C) their contents, then select one or more cells, and paste (Ctrl+V) the copied contents into them.

Location of control values:

The control value "locations" are concentrated toward the boundaries of the surface, using Greville spacing.

Changing the number of control values:

When you change the number of control values, SurfaceWorks generates a new set of control values by performing a least-squares fit to the original surface. In this way, the shape of the original surface is preserved as much as possible.

Slope and Curvature Continuity

If you want to impose slope or curvature continuity along an edge of a Tangent Boundary Surface, the bounding curve for that edge must be either a snake, a non-linear planar curve, or a point entity. Edge conditions are applied in various ways, depending on the kind of entity used to define the edge:

When the boundary curve is a snake, ring or magnet AND:

- The edge condition is G1 - the Tangent Boundary Surface will have continuous slope with (be tangent to) the parent surface of the snake, ring, or magnet.

- The edge condition is G2 - the Tangent Boundary Surface will have continuous slope and curvature with the parent surface of the snake, ring, or magnet.

When the boundary curve is a curve that lies in one plane AND:

- The edge condition is G1 - the Tangent Boundary Surface will be normal to the plane of this edge.

- The edge condition is G2 - the Tangent Boundary Surface will be normal to the plane and will have zero curvature in the direction normal to the plane.

When the boundary curve is a point (a degenerate edge) AND:

- The edge condition is G1 - the Tangent Boundary Surface will be tangent to the plane that is tangent to the two adjacent boundary curves at this corner.

- The edge condition is G2 - the Tangent Boundary Surface will be tangent to the plane and will have zero curvature in the radial direction from the corner.

Therefore, to make a Tangent Boundary Surface that:

- has continuous slope and/or curvature with an adjacent surface, use a snake along the edge of the adjacent surface as the edge for the Tangent Boundary Surface.

- ends normal to the plane of a planar edge, use a curve built in that plane or projected onto it as the edge for the Tangent Boundary Surface.

To specify edge tangency:

You specify edge tangency for a Tangent Boundary Surface by clicking the appropriate radio button on the General tab of the Insert/Edit dialog. The continuity radio buttons (G0, G1, G2) will be enabled or disabled (grayed out) depending on whether the corresponding bounding curve can support that condition.

G0: No tangency along this edge. This is the only choice available when the edge is a non-planar curve or a line (either a Line entity, or another 3D curve that happens to be straight).

G1: Continuous slope. Along an edge with G1 continuity, the first row of control values is fixed at zero (and grayed out, indicating non-editability). Available when the edge is a snake, non-linear planar curve, or a point entity.

G2: Continuous slope and curvature. Along an edge with G2 continuity, the first and second rows of control values are fixed at zero (and grayed out, indicating non-editability). Available when the edge is a snake, non-linear planar curve, or a point entity.

Changing edge continuity:

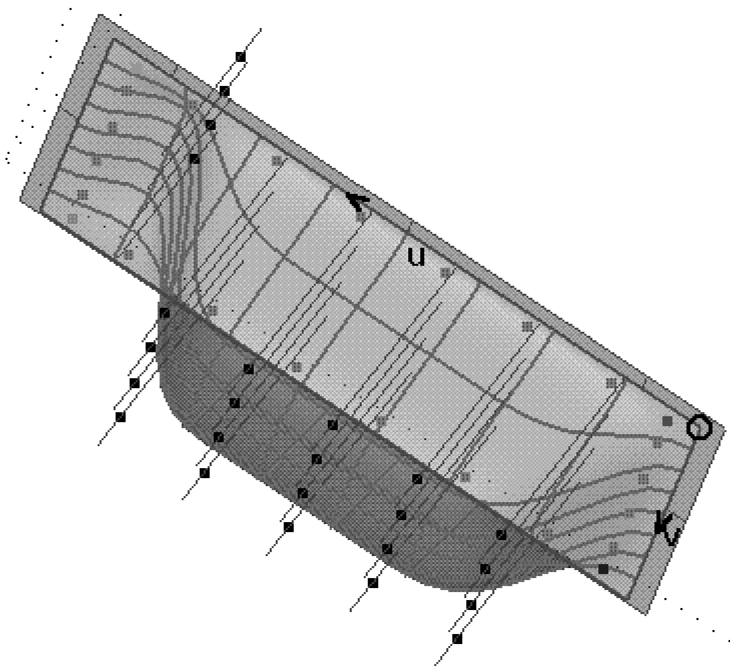
When you change edge continuity AND that change resets any control values to zero (e.g. you change from G1 to G2 and had previously set values for the second row of controls), SurfaceWorks stores the previous non-zero control values. If you then change back to a continuity setting that no longer constrains a value to zero (e.g. from G2 to G1 or G0), the previous non-zero control value will be restored.

You might find this useful, for example:

- If you had made a surface with G1 continuity, then wanted to see how it would do with G2 continuity, and finally decided you liked the G1 surface better.
- If you had changed continuity for an edge, but accidentally had picked the wrong one, so needed to reset that edge’s continuity back and change continuity for the right one.

Example: TangentBoundary2

‘surface1’ is a Tangent Boundary Surface whose four bounding curves are Projected Snake edges of the planar face that was chosen as a parent. The surface has 8 x 6 control sliders. Since G1 continuity has been selected for all edges, the outer row of sliders is fixed at zero. The other 6 x 4 controls have been used to shape the interior of the surface. G1 continuity makes the Tangent Boundary Surface tangent to the rim surface (a Trimmed Surface).



TangentBoundary2 as displayed in edit mode.

See also: Centerpoint Boundary Surface

Tangent Magnet

Parents and characteristic properties

Ring = location of tangency
u,v-offset = distance from surface

Description

The Tangent Magnet constructs a point on a surface along the direction tangent to a snake. It is highly analogous to a Tangent Point in 3-D (a point along the 3-D tangent to a curve, at a place on the curve specified by a bead). In the Tangent Magnet case, the tangency direction is really in the 2-D u,v-parameter space of the surface. Tangent Magnet is a key to constructing snakes that are tangent to each other in both u,v parameter space and in 3-D.

A Tangent Magnet is a draggable point. The tangent direction is fixed by the ring and snake parents, so only the u,v-offset (a single degree of freedom) is controlled by dragging or nudging.

Ring designates both the snake to be tangent to, and the point along the snake where the tangent direction is taken. The host surface of the Tangent Magnet is the host surface of the ring (and therefore of the snake).

u,v-offset a distance in u,v parameter space from the ring parent. This is a signed distance, positive along the positive tangent direction at ring.

Construction

The program evaluates the ring parent, obtaining the host snake and a t parameter value. Then it makes a level-1 evaluation of the snake at this parameter value, obtaining the host surface, a u,v location for the ring, and the first derivative of the u,v location. It normalizes the first derivative to obtain a unit vector U in u,v space. It multiplies U by the u,v-offset and adds the resulting 2D vector to the ring's u,v coordinates, to obtain the u,v coordinates of the TanMagnet.

Sample file Tangent Magnet.ms2

AbsRing 'r0' (bright white) is on BSnake 'n0' (bright magenta). 'r0' is used as parent for TanMagnet 'm0' (bright red); the u,v-offset is 0.200. 'r0', 'm0', and two additional AbsMagnets 'm1' and 'm2' are used as control magnets for BSnake 'n1' (yellow). Use of the TanMagnet as the second control point of 'n1' has made 'n1' durably tangent to 'n0' at 'r0'.

SubSnake 'n2' (bright green) is the portion of 'n0' from t=0 to 'r0'. SubSnake 'n2' and BSnake 'n1' are assembled into PolySnake2 'n3' (bright blue). Breakpoints command reveals only 1 degree-2 breakpoint for 'n3', at the location of 'r0'. This shows that an accurate tangency has been achieved between its component snakes 'n1' and 'n2'.

On layer 1, an image of the u,v parameter space has been constructed using a 4-point BSurf 's0' as the parameter square, and CopySnakes 'N0' and 'N1', CopyRing 'R0', and CopyMagnets 'M1' and 'M2' as images of the corresponding 3-D objects.

See also: Tangent Point

Tangent Point

Parents and characteristic properties

Bead/Ring = bead or ring. point of tangency on curve
Offset = distance of Tangent Point along tangent

Description

The Tangent Point is a point located along a tangent line to a curve or snake.

Bead/Ring serves two purposes:

1. if it is a bead, it specifies which curve the tangent is taken to. If it is a ring, is specifies which snake the tangent is taken to.
2. it specifies the position along that curve or snake where the tangent takes off

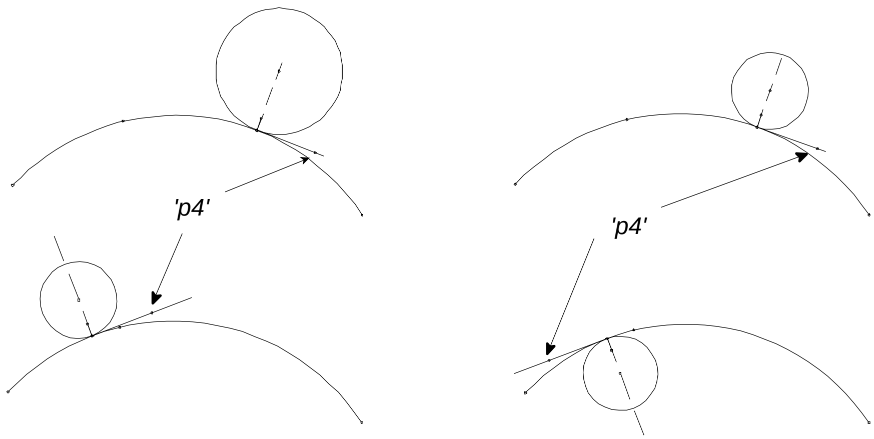
Offset specifies the (signed) distance of the Tangent Point from the bead or ring. A positive Offset is in the direction of increasing t.

Example: tanpt1

In this example, (top left figure), the Tangent Point ‘p4’ (bright red) is used in the construction of a type 'Circle: start-center-plane guide' Arc, 'c2' which is durably tangent to the type 'Arc through 3 points' Arc 'arc1'. The Tangent Point ‘p4’ (bright red) and the Point with a Point parent 'p5' are used with 'e1' as parents for the Frame 'frame1'. Point 'pt2' is aPoint with a Frame parent which is used with 'e1' to make a line on the Z-axis of 'frame1'. Arc-length Bead 'e2' is a bead on this line. Finally, 'e1', 'e2' and 'p4' are used to create 'c2'.

Since 'c2's radius is controlled by 'e2', changing the distance of 'e2' along its line changes the radius of 'c2' (top right figure). All of this geometry is linked through 'e1' so moving 'e1' moves the whole collection (bottom left figure).

If you click 'Reverse Offset' for 'p4', or change the sign of the 'e2's Location, ‘c2’ will be on the inside of ‘arc1’, still tangent to it (bottom right figure).



TanPt1.ms2: initial model (top left), 'e2' moved to arc-length = 3 (top right), ‘e1’ moved to t = 0.25 (bottom left), ‘p4’ Offset changed to -3.0 (bottom right).

See also: Point

Text Label

Parents and characteristic properties

Frame = the printing plane
Point = locates the text; see justification options
Legend = (string) the actual text to be displayed
Font = (string) Font name
Height = Text size. Real with units of length
Rel-width = a scale factor (unitless) for the character aspect ratio
Kind = 0 to 1
Justify = integer, 0 to 8 Text box position with respect to the Point parent
Reals = Variable length list of reals; can be empty. (Current maximum of 4.)

Description

The general desire is to be able to attach text labels to geometry in 3-D relational models. A primary motivation is to be able to output textual information and numerical values (such as dimensions) in 2-D drawings exported as DXF.

Frame = the printing will be in a plane parallel to the x-y plane of this frame.

Point = this is what locates the text; see justification options

Legend = (string) the actual text to be displayed; can include format specifications (see below) for formatting the values of real parents.

Font = (string) Font name

Height = Text size. Real with units of length

Rel-width = a scale factor (unitless) for the character aspect ratio

Kind = 0 to 1

Kind = 0, the text (and box) are drawn in 3D projection

Kind = 1, the text (and box) are always drawn in the plane of the screen

Justify = integer, 0 to 8 -- How is the text box positioned with respect to the Point parent?

Bottom – Left, middle, or right

Middle – Left, middle, or right

Top – Left, middle, or right

Reals = Variable length list of reals; can be empty. (Current maximum of 4.) These are formatted as parts of the message, using any format instructions in Legend.

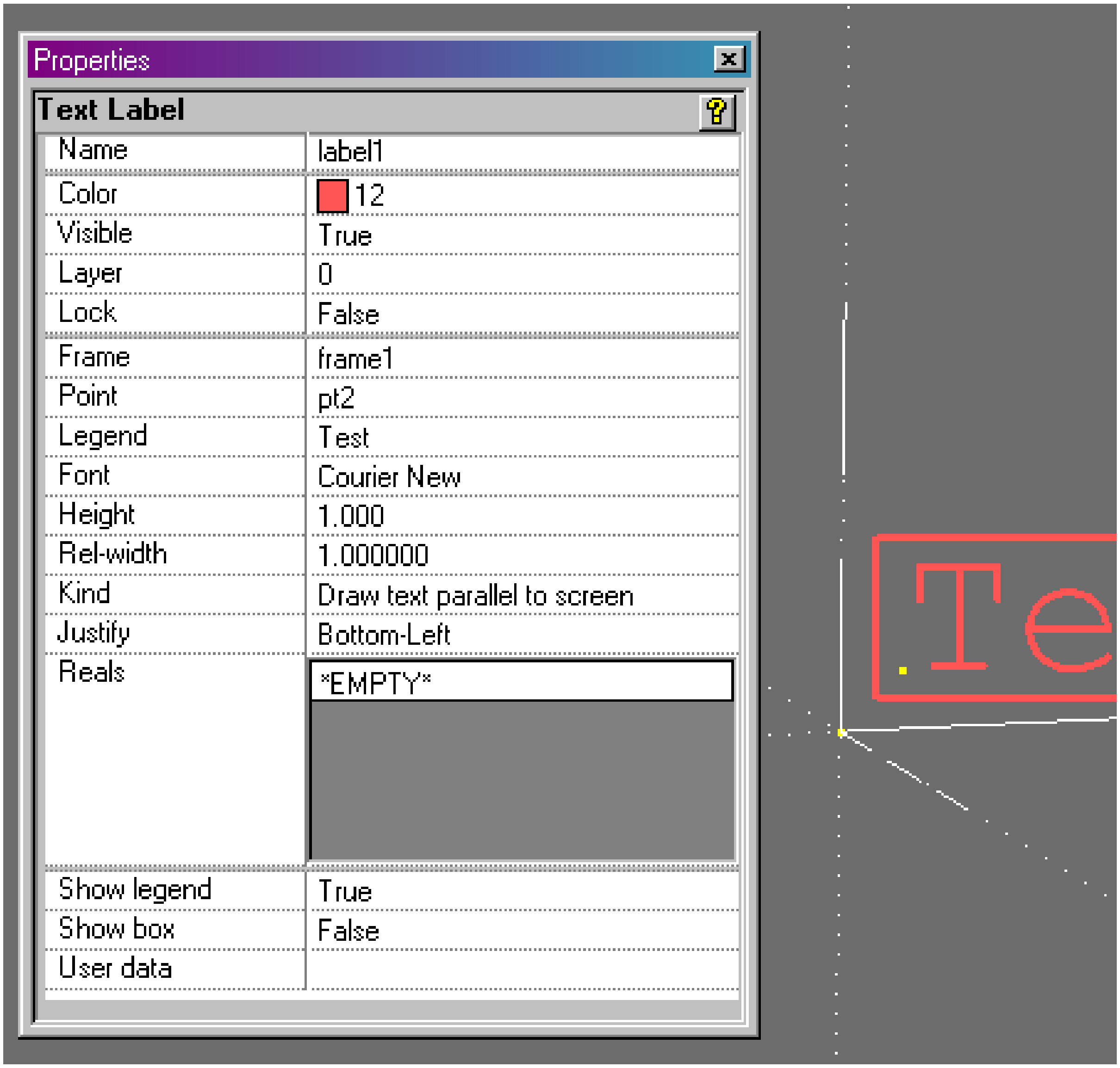
Real Formatting

A format specification similar to Visual Basic's is used. A single pound sign, '#', is used to insert an integer in a label. More #'s can be used but are not needed and will be ignored. A period followed by any number of #'s is used to insert a real number. The number of #'s following the period indicates the number of places the real number will be printed out. #'s can precede the period but are ignored for the purposes of formatting the real number.

The formatting can be further controlled by carriage returns. Use the <Enter> key to insert a carriage return. (In the model file, the character combination '\P' represents a carriage return.)

"Draw Text Parallel to Screen" issues

The benefit of drawing text parallel to the screen is that the text would be readable regardless of the viewing direction, however, there are some real issues with this. The data for export needs to be available in RGKernel, because that's where the DXF file is written. But, the kernel doesn't have any awareness of the viewing direction; and there can be various viewing directions in effect at any particular time. In the example shown below the text is constrained to be parallel to the viewing screen and a child of the visible Frame. As noted below, when this text is exported as a 3D .DXF it will revert to be drawn in a 3D projection. In the example the text will show as a projection on the frame's XY plane. By using frames in SurfaceWorks, it is possible to show text in any orientation after export to a 3D .DXF file.



Currently, all labels are written to .DXF files as if kind was set to "Draw Text in 3D Projection". This is perfectly acceptable since this is a 3D DXF file being exported and users are free to orient the text any way they want once the .DXF file is opened in AutoCAD or similar.

2D DXF export

At this time, text is not exported from SurfaceWorks to the 2D .DXF format.

Limited number of characters

SurfaceWorks puts a limitation of approximately 1000 characters on the length of the character string.

Example files

Label1.ms2 - has one Label entity, with no reals.

Label2.ms2 - has one Label entity, with 1 real. (It formats the X position of point p3.)

Label3.ms2 - has one Label entity, with 1 real. (It formats distance between point1 and point2.)

TMTrim1

Trimmed TriMesh

See Insert in the Commands chapter.

Translation Surface

Parents and characteristic properties

Stationary curve = curve entity. “stationary” parent curve
Sliding curve = curve entity. “sliding-copy” parent curve

Description

A Translation Surface is formed by sliding (translating) a copy of one parent curve (Sliding curve) along the other parent curve (Stationary curve). “Translation” implies absence of rotation, i.e., the sliding curve remains parallel to its original orientation as it sweeps out the surface.

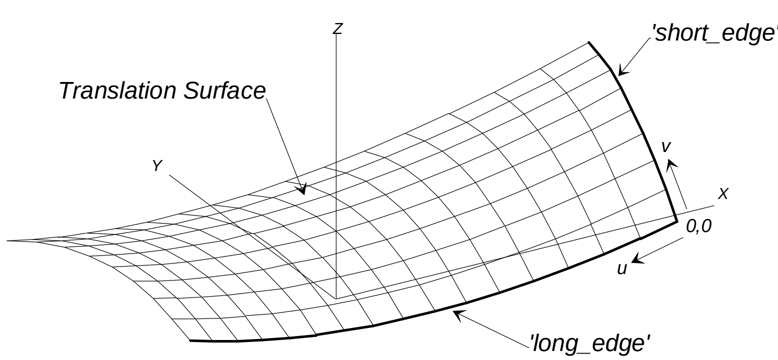
The Stationary curve in its original position is one edge (v = 0) of the surface. The u parameter of the surface is the same as the t parameter of the Stationary curve. u = 0 is the starting position of the copy of the Sliding curve, and u = 1 is its final position. The v parameter of the surface is the same as the t parameter of the Sliding curve.

You can also think of the Translation Surface as being swept out by a copy of the Sliding curve sliding along a copy of the Stationary curve. However, the situation is not completely symmetric, since Stationary curve is always the v=0 edge of the resulting surface, whereas Sliding curve can be located anywhere at all; only copies of it are used.

A general cylinder is a special case of a Translation Surface, in which either the Stationary curve or the Sliding curve is a straight line (and the other curve is a circle). If the Stationary curve and the Sliding curve are both straight lines, the Translation Surface is a plane parallelogram. If Stationary curve and Sliding curve are perpendicular straight lines, the Translation Surface is a plane rectangle.

Example: TranSurf

To form the Translation Surface ‘patch’, the curve ‘short_edge’ (Sliding curve) was swept along the curve ‘long_edge’ (Stationary curve).



TranSurf.

See also

[Sweep Surface]

See also: Sweep Surface

TriMesh

Only available through the '[MakeTriMesh]' command.

Parents and characteristic properties

degree = degree of subdivision
Points = A list of points that are the nodes.
Triangles = A list of integers, which are node indices in groups of three; each group of three is a triangle.
Breaks = A list of node indices that identify breakpoints and breaklines.

Description

This is the most basic form of TriMesh, where the nodes are explicit point entities, and the connection into triangles is explicitly specified.

A subdivided TriMesh (degree > 0) is related in shape to its coarse-mesh nodes in a way that is similar to the relationship of a B-spline Surface or NURBS surface to its net of control points. The surface generally doesn't pass through its coarse-mesh nodes, but instead acts as if it is attracted to them. TriMesh has a "net" visibility option which just shows the coarse triangulation, to help visualize this relationship. A major difference is that, while the control net of a B-Spline surface has to be organized into orderly rows and columns of control points, the TriMesh control net has a much freer topology -- it can be any valid triangle mesh.

Example TriMesh1.ms2

TriMesh 'tm' is made from 5 points, named 'p1' to 'p5' (turn on point nametags). The points are nodes 1 to 5. As you can see in the Triangles data, there are 4 triangles, with node indices: 1, 2, 3; 2, 5, 3; 5, 4, 3; 4, 1, 3. The Degree is 0, i.e., no subdivision is being applied.

Joint.ms2

A Trimesh model of a human leg. Change the degree to get better definition.

TriMesh B-spline Snake

Parents and characteristic properties

Type = B-spline degree: 1 = linear, 2 = quadratic, 3 = cubic
Periodic = Switch controlling periodic property; 0 or 1
TMMagnets/TMRings = Control points; Triangle mesh magnets or rings.
magnet1, magnet2,...magnetN

Description

A TriMesh B-spline Snake is a continuous curve embedded in a TriMesh entity and governed by a set of TriMesh Magnets as control points. The magnets must all be hosted by the same Triangle Mesh. Except for degree-1, the snake does not generally pass through its control points, but instead behaves as if it's attracted to them.

Moving one control point produces a local modification of the curve shape that is strongest in the vicinity of that control point.

At the time of Surface Works 5.0 release, there is no difference in the behavior of degree 2 and degree 3 for this entity type.

Setting the periodic property makes the snake reuse its control points in a cyclic fashion, so the snake is a closed curve (t = 0 and t = 1 are at the same location), and in fact is periodic (t and t+N are at the same location, where N is a positive or negative integer). The t=0 location on a periodic snake is generally close to magnet1, but the snake does not generally pass through magnet1.

A non-periodic TriMesh B-spline Snake does begin at magnet1 and end at magnetN, similar to a B-spline Curve or Snake.

For details about B-spline degrees and how they behave, see "B-spline entities -- Degrees".

See also: B-spline Curve, B-spline Snake

Example TMBSnake2.ms2

This model has a total of four TriMesh B-spline snake examples. The host TriMesh 'tm' is similar to LtTriMesh1.ms2, but has a subdivision degree of 2.

'N2' (yellow) is a degree-1 snake with 3 control magnets, 'G0', 'G1', and 'G2'. 'N0' (magenta) is the degree-2 snake made from the same 3 magnets.

'N3' (yellow) is a degree-1 periodic snake made from 4 control magnets 'G3' to 'G6'. 'N1' (magenta) is the degree-2 periodic snake made from the same 4 magnets.

TriMesh Copy Magnet

Parents and characteristic properties

Magnet/TMMagnet = basis Magnet or TriMesh Magnet
Surface/TriMesh = the destination TriMesh

Description

The TriMesh Copy Magnet reads the u,v position and triangle number of the basis magnet or TriMesh Magnet, and locates itself in the same triangle and at the same u,v position in that triangle on the destination TriMesh. Thus the TriMesh Copy Magnet is “slaved” to the position of its basis Magnet. The two TriMeshes must be "congruent"; that is they must have the same number of nodes and triangles, with the same indexing, order and connectivity.

Examples

TMCopyMagnet.MS2 - The parent magnets of the B-Spline Snake on the 3D Surface are copied to the Expanded TriMesh with TMCopy Magnets.

TMCopyMagnet1.MS2 - Rings along the edge of the 3D Surface are represented as TMCopy Magnets on the pattern trimesh. The TMCopy Magnets could be used as notches for pattern piece alignment.

TriMesh Copy Ring

Parents and characteristic properties

Bead/ring = basis bead or ring
Snake/TMSnake = destination Snake or TriMesh Snake
Graph = graph

Description

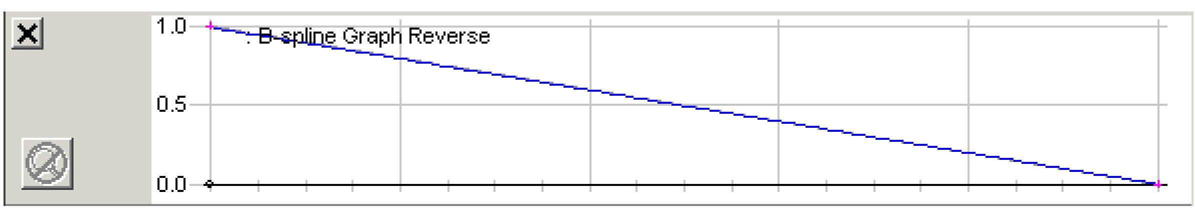
A TriMesh Copy Ring is a copy of a bead or ring from its t-location on a “source” curve or snake to a t-location on a “destination” Snake or TriMesh Snake. The source curve or snake and the destination snake are the two “environments”.

A TriMesh Copy Ring, residing on a snake or TriMesh Snake, also marks a position on the sake’s host trimesh, and so it can serve in any role where a TriMesh Magnet is needed.

In its simplest manifestation (with graph = ‘*’), a TriMesh Copy Ring is a ring on a snake, ‘N2’, which copies the t position of the basis bead or ring on its host curve or snake ‘N1’. We think of the TriMesh Copy Ring as being “slaved” to its supporting bead or ring, often functioning as a “remote control”.

The Graph support greatly enriches the behavior of a TriMesh Copy Ring by permitting a transformation of the parameter between the source curve and destination curve. The default graph ‘*’ in this context signifies no transformation, i.e., h(t) = t. A more complex graph produces a more complex motion or response of the TriMesh Copy Ring. Since there are few limits on the complexity of a graph you can specify practically any relationship between the positions of the two beads.

A simple graph example that is frequently useful is the type-1 B-spline Graph with values {1, 0}, i.e., h(t) = 1 - t . We usually give this graph the name ‘reverse’, but now it is available as a default system entity seen above as '*10'. If introduced into the above example of two snakes, it makes the TriMesh Copy Ring go to the complementary position 1 - t, i.e. it traverses ‘N2’ from 1 to 0 as the basis bead or ring goes from 0 to 1.



Examples

TMCopyRing.MS2 - The TMCopy Ring is parented by a ring on a parametric surface.

TriMesh Copy Snake

Parents and characteristic properties

Snake/TMSnake = parent Snake or TriMesh Snake
Surface/TriMesh = destination surface for the Copy Snake
Direction = orientation direction

Description

The TriMesh Copy Snake is a copy of a parent Snake or TriMesh Snake onto a different surface or TriMesh. Each point of the TriMesh Copy Snake has the same triangle number and u,v position on its parent TriMesh as has the corresponding point of the parent snake on its host TriMesh.

Relabel is used to relabel the curve. The program's default Relabel '*' produces the "natural" labeling, in which the parameter t labeling corresponds point by point to that of the basis snake.

Snake/TMSnake is the snake to be copied.

Surface/TriMesh can be a surface or any kind of TriMesh entity, but it must be congruent to the source TriMesh; that is, it must have the same number of nodes and triangles, with the same indexing, order and connectivity.

Direction is the orientation angle for the copy, normal or reversed.

Examples

TMCopySnake.MS2 - The TMCopy Snake is parented by a TMProjected Snake on a TriMesh.

TriMesh Edge Snake

Parents and characteristic properties

Type = Counterclockwise (0) or Clockwise (1)
TMMagnet1 = TriMesh Magnet/Ring for the t = 0 end
TMMagnet2 = TriMesh Magnet/Ring for the t = 1 end

Description

A TriMesh Edge Snake is a curve that follows the boundary of a TriMesh, between two TriMesh Magnets. The two magnets must be hosted by the same TriMesh entity; they must both lie on the boundary; and if the TriMesh has multiple boundaries, the two magnets must be on the same boundary.

Two boundary magnets divide a boundary loop into two parts. Type determines which of those parts becomes the TriMesh Edge Snake.

Example TMEdgeSnake1.ms2

This model has two TriMesh Edge Snake examples. The host TriMesh 'tm' is similar to the Trimesh1.ms2 example, but has a subdivision degree of 2. The two TriMesh magnets 'G0' and 'G1' (bright red) lie on the boundary of 'tm'. Edge Snake 'N0' (magenta) goes from 'G0' to 'G1' in the counterclockwise direction. Edge Snake 'N1' (yellow) is made from the same two magnets, in the same order, but, being type-1, goes in the clockwise direction.

TriMesh Intersection Snake

Parents and characteristic properties

Type = 0 to 1
TMMagnet = a TriMesh magnet or ring

Mirror/Surface = plane, frame, surface, Line or point. Defines a family of potential cutting planes or surfaces.
Point = point entity. Specifies the actual cutting plane or surface.

Description

A TriMesh Intersection Snake is a curve on a TriMesh, located where a real or virtual cutting plane or surface intersects the TriMesh.

The TriMesh Magnet parent serves three roles:

It specifies which TriMesh the Intersection Snake is cut on (i.e., the TriMesh that is host to the TriMesh Magnet)

If there are multiple intersections, the one closest to the TriMesh Magnet is selected.

The end of the selected intersection that is nearest to the TriMesh Magnet becomes the t = 0 end of the Intersection Snake.

The surface doing the cutting can be the actual plane, frame (its x,y-plane) or surface specified by Mirror/Surface, if Point lies in that plane or surface; or it can be a virtual cutting surface offset from Mirror/Surface and passing through Point. (For details, see "Specifying the cutting surface".)

The Type property is significant only when the intersection is a closed curve. Then it is used to determine the clockwise/ counterclockwise orientation of the Intersection Snake.

Example TMIntSnake1.ms2

'tm' is a TriMesh similar to TriMesh1.ms2. The parents of TriMesh Intersection Snake 'N0' are:

TriMesh Magnet 'G0' (red)

System plane *Y=0

Bead 'e0' (bright white)

'G0' specifies the host TriMesh for the snake ('tm', host of 'G0'), and selects which end of the cut will be the t = 0 end of the snake. The cut is with a plane parallel to *Y=0, passing through 'e0'. (2-point Plane 'a0' is included in the model to show the cutting plane. but is not directly involved in the Intersection Snake.)

TriMesh Magnet

Parents and characteristic properties

Triangle mesh = Host TriMesh
Triangle no. = A triangle index in the host TriMesh
u,v = Barycentric coordinates in the selected triangle

Description

A TriMesh Magnet is a point located on a TriMesh by {triangle index, u, v} location in the coarse mesh. It stays in the same relative position when the host TriMesh is subdivided.

Dragging is implemented in SurfaceWorks.

Example TMBSnake2.ms2

This model has 7 TriMesh Magnets supporting two TriMesh B-spline Snakes.

See also: TriMesh Ring

TriMesh Projected Magnet

Parents and characteristic properties

Point = Point entity to project
TMMagnet/ TriMesh = designates the host TriMesh that the TriMesh Projected Magnet will lie on.
Mirror/TriMesh. = Plane, line, point or TriMesh entity.

Description

A TriMesh Projected Magnet is a point located on a Trimesh by projection of another Point along a straight line.

When Mirror/TriMesh is a plane, the projection line is normal (perpendicular) to the plane.

When Mirror/TriMesh is a line, the projection line is normal (perpendicular) to the line.

When Mirror/TriMesh is a point, the projection line radiates from the point.

When Mirror/TriMesh is a TriMesh (it has to be the same TriMesh that the TriMesh Projected Magnet is to lie on) plane, the projection line is along the local normal direction to the TriMesh; the TriMesh Projected Magnet will be at the closest point of the TriMesh to Point.

TMMagnet/TriMesh can be either a TriMesh Magnet or Ring, or a TriMesh entity.

When TMMagnet/TriMesh is a magnet or ring, it serves two purposes:

1. It specifies which TriMesh the TriMesh Projected Magnet is to lie on -- the host TriMesh of the magnet or ring.

2. It gives a starting location for the iterative search for the projection point. In the case of multiple intersections of the host TriMesh with the line of projection, the TriMesh Projected Magnet will (usually) be made at the intersection nearest to the starting location (in terms of distance measured within the TriMesh, not 3-D distance).

When TMMagnet/TriMesh is a TriMesh, it specifies directly which TriMesh the TriMesh Projected Magnet is to be hosted on. In case of multiple intersections of the projection line, the starting location for the search will be at triangle 1 of the host TriMesh, and the search will usually converge to the intersection closest to triangle 1.

Example TMProjMagnet1.ms2

TriMesh Projected Magnet 'G1' (bright red) is the vertical projection of Point 'P0' onto the Surface TriMesh 'I0', using the system plane *X=0 for the mirror.

TriMesh Projected Magnet 'G2' (bright white) is the normal projection of Point 'P0' onto the Surface TriMesh 'I0', by using 'I0' as the Mirror/TriMesh parent.

TriMesh Projected Snake

Parents and characteristic properties

Curve = curve entity
TMMagnet/TriMesh = TriMesh magnet or ring, or TriMesh entity. Designates the TriMesh on which the Projected Snake will lie.
Mirror/TriMesh = plane, frame, line, point, or TriMesh entity
Draft Angle = Draft Angle (degrees) -- Not currently implemented

Description

A TriMesh Projected Snake is a projection of Curve onto a TriMesh. Each point of Curve is projected along a straight line to locate the corresponding point on the Projected Snake. The projection can be done with any valid type of mirror, or can be normal to the TriMesh host:

When Mirror/TriMesh is a plane, the projection lines are normal (perpendicular) to the plane.

When Mirror/TriMesh is a Line, the projection lines radiate from the Line and are normal (Perpendicular) to the Line.

When Mirror/TriMesh is a point, the projection lines radiate from the point.

When Mirror/TriMesh is a TriMesh (it must be the same TriMesh the projected Snake is to lie on), the projection is along the local normal direction to the TriMesh. Each point of the Projected Snake will be at the foot of a line from the corresponding point on Curve dropped normally to the TriMesh. (Usually this is the closest point on the TriMesh.)

TMMagnet/TriMesh specifies the TriMesh the Projected Snake is to lie on. When TMMagnet/TriMesh is a TriMesh magnet or ring, it serves two purposes:

It specifies which TriMesh the Projected Snake is to lie on: the TriMesh that is host to TMMagnet.

It provides a starting location on the TriMesh for the iterative search that locates the first point (the t=0 end) of the Projected Snake. This is especially important in case of multiple intersections of the TriMesh with the projection line; then the location of TMMagnet will help to select the correct one of two or more possible Projected Snakes.

Whenever you create a TriMesh Projected Snake, we recommend you use a magnet or ring for the TMMagnet/TriMesh parent, locating it reasonably near where the Projected Snake is to begin.

When TMMagnet/TriMesh is a TriMesh, it specifies directly the host triangle mesh for the Projected Snake. The search for the intersection point at the t = 0 end of the snake will start at the first triangle of the TriMesh.

Example TMProjSnake2.ms2

'tm' is a dome-shaped TriMesh entity with 7 nodes, 6 triangles, and 2 degrees of subdivision. Curve is a B-spline Snake 'n0' (magenta) drawn on the B-spline Surface 's0'. TriMesh Projected Snake 'N0' (yellow) is the vertical projection of 'n0' onto 'tm', using the system plane *Z=0 as mirror. TriMesh Magnet 'G1' (bright red) identifies the host TriMesh for the Projected Snake (its host, 'tm'), and provides a starting location for the projection of its t = 0 end.

TriMesh Ring

Parents and characteristic properties

TMSnake = A TriMesh Snake entity
t position = location along snake

Description

A TriMesh Ring is a point located by t parameter value along a TriMesh Snake. It can serve to mark a location along the snake; it also serves as a location on the host TriMesh.

Dragging of TriMesh Rings is implemented in Surface Works.

Example TMBSubSnake1.ms2

This model has four TriMesh Rings (bright blue) marking locations on two different TriMesh B-spline Snakes (magenta) for the ends of SubSnakes (yellow and bright cyan).

TriMesh SubSnake

Parents and characteristic properties

Degree = blending function B-spline degree, 1 to --
Direction = Normal or Reversed
TMRings = ring1, ring2, ... ringN

Description

A TriMesh SubSnake is a portion of a parent TriMesh snake, between ring1 (t = 0 on the SubSnake) and ringN (t = 1 on the SubSnake). In most cases, 2 TriMesh Rings will be sufficient (N = 2). When more than 2 rings are used as parents (N > 2), the intermediate rings control the labeling of the SubSnake, using B-spline blending in a fashion similar to SubCurve.

Notice that if ring1 is at a larger t-value on the parent snake than ringN, the t parameter on the SubSnake will run in the opposite direction from the t parameter on the parent snake.

The TriMesh SubSnake is hosted on the TriMesh that is host to its parent TMRings. The TMRings must all be on the same TriMesh snake (the parent snake). Predefined rings *0 and *1 can be used for any of the TMRings, **except** at least one of the TMRings has to be an actual bead or ring, in order to specify the parent snake.

Degree is the B-spline degree for the blending functions: 1 = linear, 2 = quadratic, 3 = cubic, etc. As usual, if the number of parents N is less than Degree+1, the splines are automatically demoted to degree N+1.

The Direction property is active only when the parent snake is periodic. Then there are two possible ways to get from ring1 to ringN -- clockwise and counterclockwise. When Direction is Normal, the subsnake will use the part that goes around in the host snake's positive t direction; when Direction is Reverse, it will use the other part.

Example TMBSubSnake1.ms2

This model has three example TriMesh Subsnakes.

'N2' (yellow) is between TriMesh Rings 'W0' and 'W1' (bright blue) on TriMesh B-spline Snake 'N0' (magenta).

'N3' (yellow) is between TriMesh Rings 'W2' and 'W3' in the Normal direction, i.e. its t parameter runs in the same direction as the periodic parent snake, TriMesh B-spline Snake 'N1' (magenta).

'N4' (bright cyan) is also between TriMesh Rings 'W2' and 'W3', but is in the Reverse direction, i.e. its t parameter runs counter to the t parameter of the parent snake 'N1'.

Trimmed Surface

Parents and characteristic properties

Number of triangles = number of triangles for used for shaded image and export
Magnet = magnet or ring. identifies the portion of the parent surface which is to be KEPT.
Bounding snakes = snake. snake1 . . . snakeN
Type of triangulation = pick one of three types

Description

A Trimmed Surface is a non-parameterized portion of a parent surface bounded by snakes and/or edges. It must be a connected region (i.e. two disjoint portions of a surface will not be recognized as a single Trimmed Surface). A Trimmed Surface is bounded by one or more closed loops of snakes and/or edges. It can have any number of “sides” and any number of “corners” (this is one of the major reasons you’d use a Trimmed Surface rather than a SubSurface), or no corners (e.g. with a smooth closed boundary). It can have holes. As a non-parameterized surface, a Trimmed Surface cannot be used as a parent surface for other surfaces, e.g. Fitted Surfaces or Mirrored Surfaces, but can be a parent to Trimeshes. (If you need a parameterized surface and can do without holes and a boundary defined by just 2 or 4 snakes, try using a SubSurface instead.)

For a comparison of the Trimmed Surface and SubSurface entities, see "Comparison with trimmed surfaces" on page 405.

snakes1 ... snakeN are the boundaries for the Trimmed Surface. If a snake has a durable connection with an edge, that portion of the edge which connects to the next snake in the bounding set will automatically be included as a trimming boundary. All trimming snakes must all belong to the basis surface. The keeper portion is the region inside the outer loop which contains the parent magnet and outside any inner loops. The presence/absence of holes determines how many loops of snakes are needed:

When there are no holes, only one loop of snakes/edges are needed, to define the outer boundary of the Trimmed Surface “Example 1: TrimSurf1”.

When the Trimmed Surface incorporates one or more holes, each hole must be bounded by a loop of snakes, and the outer boundary needs a loop as well Fig. 1, “Example 2: TrimSurf2_D2HLink_90249” and “Example 3: TrimSurf3_D2HLink_90250”. In many cases no outer loop snakes are needed for this case. The edges will be sufficient and the placement of the parent magnet (keeper) will determing the location of the trim.

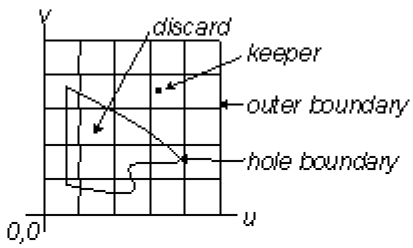


Fig. 1. u,v plane, Trimmed Surface bounding snakes, and keeper and discard regions. The keeper region is inside the outer boundary and outside any inner (hole) boundaries.

Within a loop, the snakes must be selected in order — one end of the first snake must touch one end of the second snake, etc., and the last snake must connect back to the first snake.

The loops of snakes can be picked in any order: boundary loop, then hole loop(s), or vice versa; or first hole loop, boundary loop, second hole loop; etc.

Magnet/Surface identifies the parent surface of the Trimmed Surface:

> When Magnet/Surface is a surface, it directly specifies the parent surface (surface to trim).
> When Magnet/Surface is a magnet, it indirectly specifies the parent surface (surface to trim).

Number of triangles is an integer between 10 and 10,000 that specifies the number of triangles used to produce the shaded image of the Trimmed Surface and for export of the surface as a triangle mesh. The initial SurfaceWorks default is 200.

NEW No. of triangles = 0 option

If the number of triangles is specified as 0, it uses the Trimmed Surface divisions and subdivisions to calculate the number of triangles wanted, and uses NU / NV for aspect ratio.

Type of triangulation is one of three types:

- Ignore trimming loop divisions
- Use trimming loop divisions
- Use trimming loop divisions and subdivisions

Default is "Ignore trimming loop divisions".

The effect that the number of triangles has on the outcome of the surface is controlled by the Type of triangulation chosen. If the Type of triangulation is **Ignore trimming loop divisions**, (the default setting), only the Number of triangles is used to control the division of the surface into triangles. If the Type of triangulation is **Use trimming loop divisions**, the division into triangles is controlled by a combination of the

Number of triangles property and the divisions of the parent bounding snakes. The bounding snake divisions control the triangles along the edges of the Trimmed Surface (subdivisions are ignored). Number of triangles controls the triangles used to fill the interior of the surface. If the Type of triangulation is **Use trimming loop divisions and subdivisions**, the division into triangles is controlled by a combination of the Number of triangles property and the divisions and subdivisions of the parent bounding snakes. The bounding snake divisions and subdivisions control the triangles along the edges of the Trimmed Surface. Number of triangles controls the triangles used to fill the interior of the surface.

You can see the triangles by turning on the Triangles visibility option for the Trimmed Surface in the Property Manager— check the Triangles box and uncheck the u-constant and v-constant boxes). The goal is to have approximately uniformly sized triangles (Fig. 2). If the triangles along one or more edges are much smaller (Fig. 5) or larger (Fig. 4) than the triangles in the interior, you'll need to make sure **Use trimming loop divisions** or **Use trimming loop divisions and subdivisions** is set and adjust the bounding snake divisions and / or subdivisions (**Edit>Definition>Advanced** tab). The SurfaceWorks distribution (/ Examples directory) includes the example file FlattenTrim1, which we used for making the images below. We encourage you to experiment with it yourself.

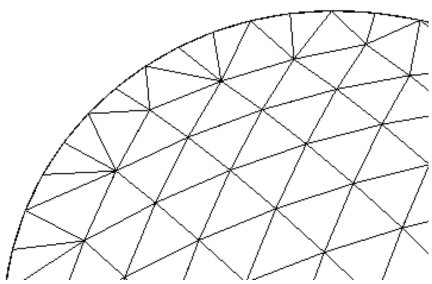


Fig. 2. Number of edge triangles okay.

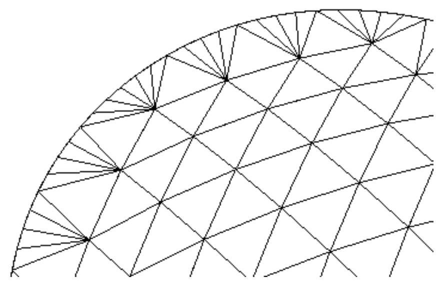


Fig. 3. Number of edge triangles also okay.

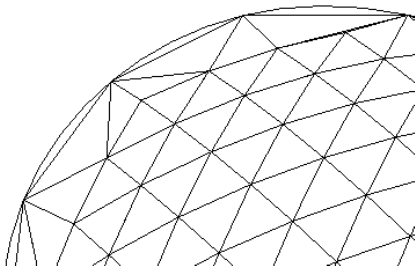


Fig. 4. Too few triangles along edges.

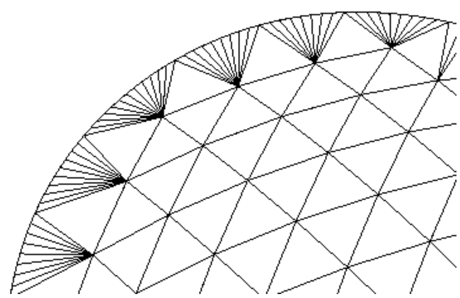


Fig. 5. Too many triangles along edges.

Recommendations

Our recommendation is to set Type of triangulation to **Ignore trimming loop divisions** and use between 200 and 400 triangles per surface. It really is easiest to work with the triangles displayed. For Number of triangles use on the order of 200 triangles if the surface has only a small amount of curvature or closer to 400 triangles if the surface has high curvature.

Then look at the triangles along the edges and adjust the divisions of the bounding snakes, if necessary, so that the edge curves are adequately resolved (Figs. 2 and 3), but there is not a large number of narrow, “slivery” triangles (Fig. 5).

Flattener (Optional Add-on) and Trimmed Surfaces

In the event that there is difficulty with flattening a Trimmed Surface using the Flattener and you have tried the suggestions as outlined above, try changing Type of triangulation to **Use trimming loop divisions** or **Use trimming loop divisions and subdivisions**. Then look at the triangles along the edges and adjust the divisions and / or subdivisions of the bounding snakes, if necessary, so that the edge curves are adequately resolved, but there is not a large number of narrow, “slivery” triangles. Follow the usual procedure for surface flattening and see if it now succeeds.

Trimmed Surface Limitations

Since a Trimmed Surface is not a parametric surface in itself, it cannot be used as a parent surface for other surface entities. (While you can put magnets on a Trimmed Surface, they are positioned according to the u,v of the parent surface.) You can join a Trimmed Surface to other entities via its boundary snakes (e.g. another surface which shares one of the snakes as a common edge; a ring on one of the bounding snakes).

Trimmed Surfaces in File Exports

Trimmed Surfaces are handled in only some 3D file exports. Currently, they are included in IGES and 3D: DXF Wireframe, 3DA Wireframe, and POV Triangles files.

Example 1:

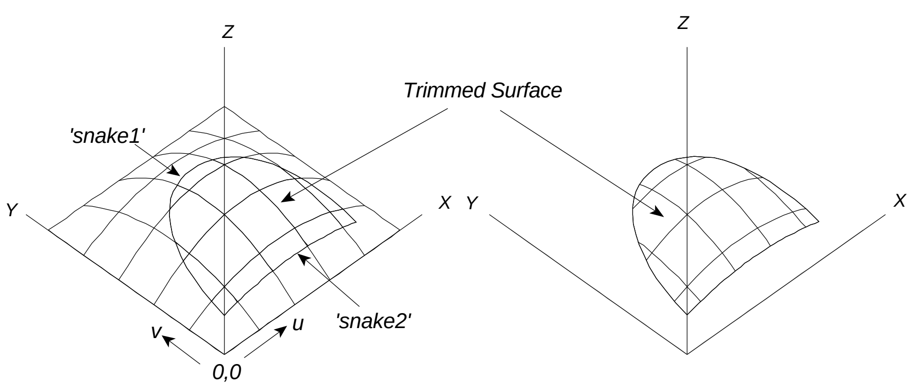


Fig. 6. TrimSurf1.

The parent surface is a B-spline Lofted Surface (‘s1’; gray); it is identified directly by Magnet/Surface = ‘s1’. Snakes ‘snake1’ (magenta) and ‘snake2’ (red) form the outer boundary for the Trimmed Surface (green).

Example

This Trimmed Surface (green) has a hole in it. Its parent Magnet/Surface is the Translation Surface ‘s1’ (gray) created from imported parent curves ‘xline1’ and ‘xcurve2’.

The outer boundary for the Trimmed Surface is formed by the Edge Snakes ‘outer1’, ‘outer2’, ‘outer3’, and ‘outer4’ (all magenta; the four edges of the parent surface). The hole boundary is a projection of the imported Conic Curve ‘xcurve1’. We projected ‘xcurve1’ onto ‘s1’, creating the Projected Curve ‘hole’.

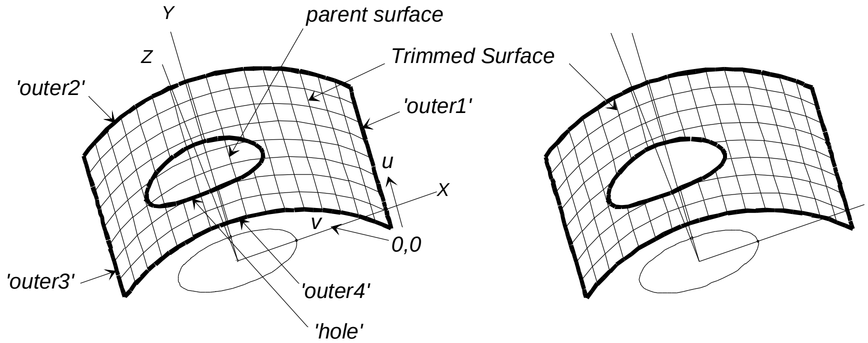


Fig. 7. TrimSurf2.

Example 3: TrimSurf3

This Trimmed Surface (bright green) has two holes in it. The parent surface is a B-spline Lofted Surface (‘s1’; gray); Magnet/Surface is the magnet ‘m2’ (it’s also a control magnet for the snake ‘hole1’).

The outer boundary for the Trimmed Surface is formed by the Edge Snakes ‘outer1’ through ‘outer4’ (all magenta; the four edges of the parent surface). The holes are each bounded by a loop composed of a single B-spline Snake which begins and ends at the same place: ‘hole1’ (blue) and ‘hole2’ (red).

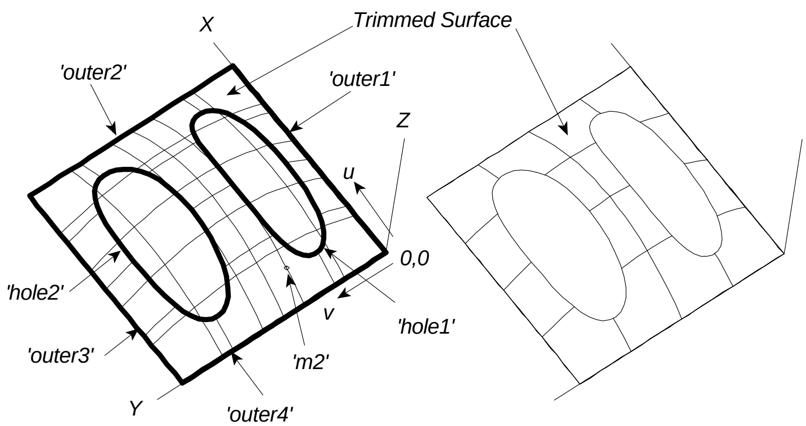


Fig. 8. TrimSurf3.

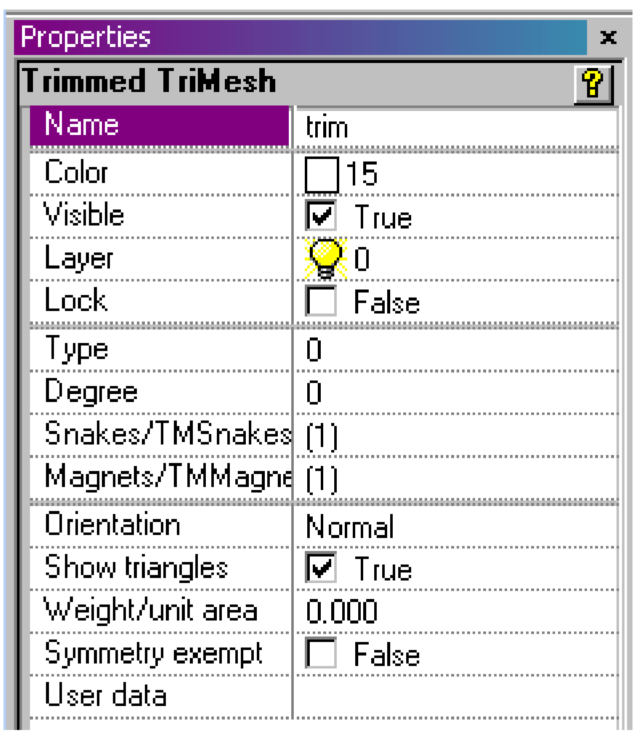
See also: SubSurface

Trimmed Trimesh

A Trimmed Trimesh is a portion of a triangle mesh enclosed by a boundary of Trimesh snakes. (This is highly analogous to a Trimmed Surface.)

Characteristic data

Name -- as with any entity
Color -- as with any graphic entity
Visible -- true / false
Layer -- as with any entity
Lock -- as with any entity
Type – 0–3 changes triangle distribution
Degree -- Coarseness of triangles
TMSnakes -- Bounding snakes off the trimesh boundary
TM Magnets -- Indicate which portion of the trimesh to keep
Orientation – as with any trimesh entity
Show Triangles -- Off shows boundary only
Weight/Unit Area -- as with any graphic entity
Symmetry Exempt -- as with any graphic entity
User data -- as with any entity



There are currently 9 types:
0 = the old TMTrim1
1, 2, 3 = the new “wide fence” kind, with fence width 0.20, 0.40, and 0.60 times the local average size of base-trimesh triangles
4, 5, 6 = the same 3 fence widths, but showing the fence triangles only

7, 8, 9 = the same 3 fence widths, but showing the tiling triangles only
Kinds 4-9 are for debugging purposes, and have no other known utility.

The boundary TMSnake divisions and subdivisions are used in generating the coving triangles. The snakes should have subdivisions that are roughly similar in size to the base-trimesh triangles they are crossing. Too few subdivisions on the snakes can cause triangulation failures.

UVSnake

Parents and characteristic properties

Type = u=constant or v=constant
Reverse snake orientation (when checked) = reverses the direction of the snake's t parameter
Magnet = magnet or ring

Description

The UVSnake is a special line snake traversing a surface at a constant value of the u or v parameter. It is one of the family of u=constant or v=constant parameter lines on the surface.

Type is either u=constant or v=constant.

The t parameter of the UVSnake runs in the same direction as either the u or v parameter of the surface. To reverse the UVSnake's t-orientation, check the “Reverse snake orientation” box.

Magnet identifies the surface and also provides the particular u or v value for the parameter that is constant along the snake. That is, Magnet specifies one point on the surface that the snake passes through. The UVSnake always goes all the way across the surface, from one edge to the opposite edge.

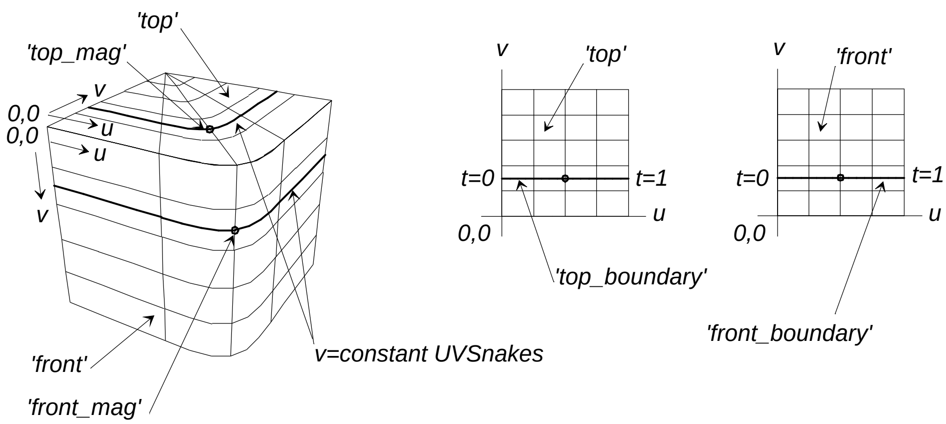
A u=constant snake could also be formed by a Line Snake joining two magnets at surface parameters (u, 0) and (u, 1). Similarly, a v=constant snake could be formed by a Line Snake joining two magnets at surface parameters (0, v) and (1, v). However, when a snake needs to have a constant surface parameter value there are several advantages in using a UVSnake:

- you need to make only two entities, rather than three
- you can drag a UVSnake by moving only one magnet
- the properties of staying at constant u or v, and traversing the entire surface from edge to edge are durable, not subject to location of magnets

Relabel is used to relabel the snake. The program's default Relabel'*' produces the "natural" labeling for the UVSnake: for a u=constant UVSnake, t is equal to v; for a v=constant UV Snake, t is equal to u. In other words, the surface parameter not held constant is the parameter along the curve.

Example: UVSnake

‘top’ is a C-spline Lofted Surface, and ‘front’ is an adjacent Translation Surface; they share a common boundary along the curve ‘rim’. ‘top_boundary’ (blue) is a v=constant UVSnake running through the Magnet ‘top_mag’ (located on ‘top’ at u = .5, v = .3), thus ‘top_boundary’ is the v = .3 line on ‘top’. ‘front_boundary’ (magenta) also is a v = constant UVSnake. It runs through the Magnet ‘front_mag’ (located on ‘front’ at u = .5, v = .3), thus making ‘front_boundary’ the v = .3 line on ‘front’.



UVSnake1.

See also: Line Snake, Edge Snake

Variable

SurfaceWorks has a profound new capability for generation of parametric families of geometric models, in a new class of real-valued objects: the Variable entity type.

Constants and variables

There are two ways to specify a real number in the syntactic definition of an object: with a constant value, or a variable. Variables are objects and carry a value that lies within a specified range.

Constants

Constants are 1.23, -0.7, 12e-8, +98, etc. Although they have no units they are always considered to carry the adequate unit needed at the particular location where they are used.

For instance in

```
AbsPoint P 1 1 / 1 2 3 ;
```

1, 2 and 3 are considered as having length units.

In

```
AbsMagnet M 1 1 / S 0.123 0.456 ;
```

the u and v parameter values 0.123 and 0.456 are considered to be unitless.

Variables

Variables carry a value; they can optionally have units (otherwise they are unitless), and take their value only within a certain range.

Basic syntax

The simplest syntax for a variable is:

Variable *name* **/** *value* **;**

For instance

Variable x / 177.69 ;

Range

If the variable is only allowed within a certain range then the boundaries of the range may be specified in order:

Variable *name* **/** *value* **(***min***,***max***) ;**

If either min or max are not specified then it is meant that the lower or higher bound in not specified (or, equivalently, is minus or plus infinity, respectively).

```
Variable x / 177.69 (0,300) ;
Variable x / 177.69 (,300) ;
Variable x / 177.69 (0,) ;
Variable x / 177.69 (,) ;
```

The last case is equivalent to the case where the range is not specified. You can create a one-value variable in the following way:

```
Variable x / 177.69 (177.69, 177.69) ;
```

Units

Variables may carry units, which are not specified by their names, but by their dimensions with respects to the fundamental units of the SI / MKSA system:

L	length
M	mass
T	time
I	intensity of electric current

For instance, velocity has unit LT^{-1}, acceleration, LT^{-2}, energy $ML^{2}T^{-2}$, etc.

The units of a variable can be specified with any combination of L, M, T and I, with the exponent introduced with '^':

```
Variable height / 100 L^1 ;
Variable mass   / 80 M^1 ;
Variable speed  / 90 L^1 T^-1 (0,300) ;
```

An exponent of 0 is equivalent to no unit of this type; if the exponent is one, it can be omitted; repeating the same unit amounts to multiplying it:

```
Variable x      / 0 M^0 ;
Variable height / 100 L ;
```

```
Variable area   / 100 L L ;
```

Complete syntax

Variables can also have user attributes and be read-only (locked). The complete syntax for a variable is:

'Variable' *name* { *attr*, } { '**R:1**' } '/' *value* { *units* } {*range* }';'

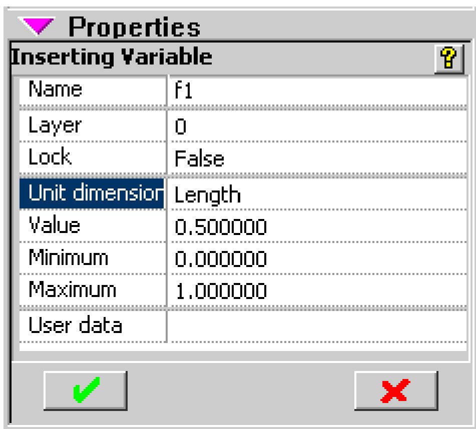
For instance:

```
Variable length A:Usage("profile") / 10 L (0,152) ;
```

User interface

Figure 1 shows the Property Manager interface to Insert and Edit a Variable Object.

Fig. 1



When you are editing a real value in the data for any object, you can use an appropriately dimensioned real object (Variable or Formula) instead of a constant. For example, creating or editing a RadiusArc, one required data element is the Radius. When you select this item a constant or a real object can be entered. Figure 2 shows a Radius Arc with the Radius field active and a radius of 1. To enter a variable, go to the Surfer View and choose the appropriate Variable. (Figure 3) Figure 4 shows Variable 'ArcRadius' as a support, which has a value of 0.500.

Fig.2	Fig.3

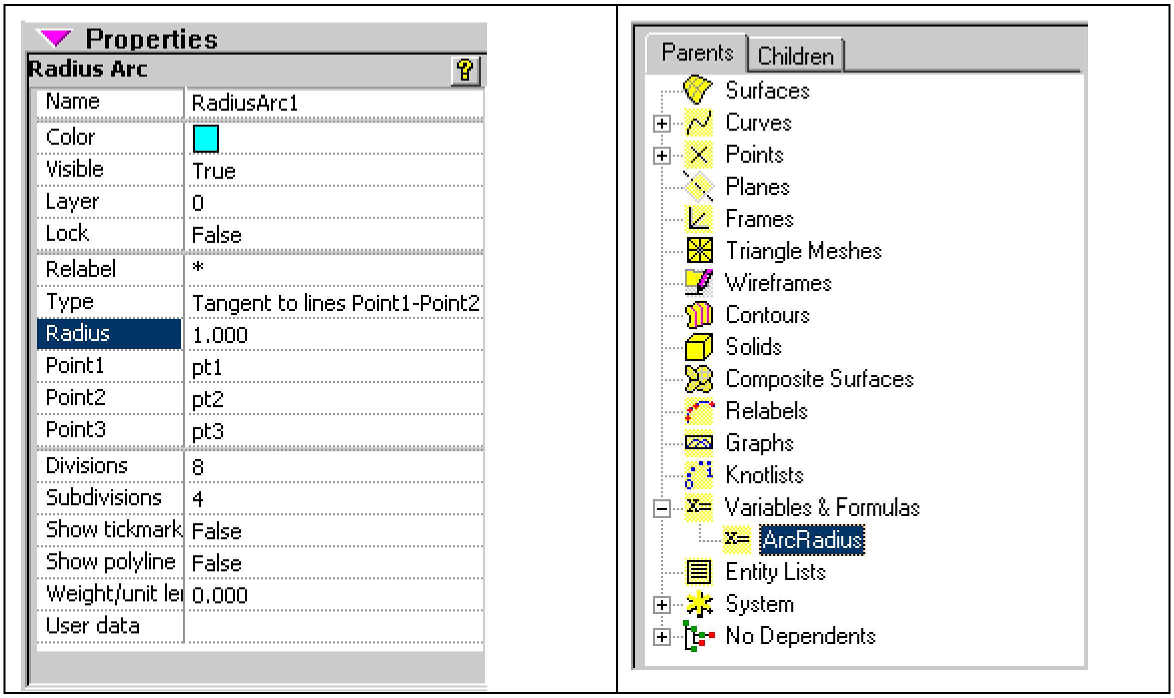
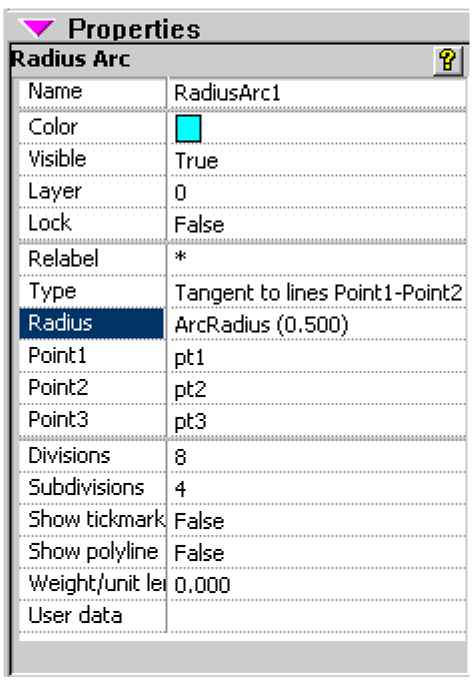


Fig. 4



Variables as supports

Variables can be supports of all objects depending on a real value, and wherever an object regardless of its type can be a support, such as in an object list. Therefore they can be supports of points (as coordinates, offsets, angles, t-, u- and v-parameter), curves and surfaces (via knot lists and weights), etc.

```
Variable A / 10 L ;
Variable B /  5 L ;
Variable C / 30 L ;
Variable D / 25 L ;
AbsPoint P1  14  1 / 0 0 0 ;
AbsPoint P2  14  1 / A 0 0 ;
AbsPoint P3  14  1 / A B B ;
AbsPoint P4  14  1 / C B B ;
AbsPoint P5  14  1 / C 0 0 ;
AbsPoint P6  14  1 / D 0 0 ;
Variable k0 / 0.25 ;
Variable k1 / 0.75 ;
KnotList kl / { 0 k0 0.5 k1 1.0 } ;
Variable w0 / 1 ;
Variable w1 / 3 ;
NURBCurve spline0 11 5 64x1 / * 2 kl { P1 1 P2 w0 P3 w1 P4 1 P5 1 P6
w0 } ;
```

Note that since a variable is defined by three real numbers, these can be replaced by variables or formulas: you can for instance have a variable with a variable range:

```
Variable min / 0 ;
Variable max / 24 ;
Variable x   / 7 (min,max) ;
```

In this case the objects used as value and range bounds must all carry the same units (as always, constant are assigned the units that match).

Interaction with object transformations

Object transformations are rotations, translations and scaling.

Scaling of variables

Variables having a non-null length dimension will be scaled. For instance, in case of uniform scaling of 10

```
Variable radius         / 100 L ;
Variable area     / 3 L^2 ;
Variable curvature      / 1 L^-1 ;
Variable kineticEnergy  / 4.2 M L^2 T^-2 ;
```

will become

```
Variable radius         / 1000 L ;
Variable area     / 300 L^2 ;
Variable curvature      / 0.1 L^-1 ;
Variable kineticEnergy  / 420 M L^2 T^-2 ;
```

The lower and upper bound of the range get scaled only if they are not $+\infty$ or $-\infty$.

```
Variable area  / 3 L^2 (1,5) ;
Variable area  / 3 L^2 (1,) ;
Variable area  / 3 L^2 (,5) ;
```

becomes

```
Variable area  / 300 L^2 (100,500) ;
Variable area  / 300 L^2 (100, ) ;
Variable area  / 300 L^2 (,500) ;
```

Scaling of objects depending on variables

Let

```
Variable d  / 9 L  ;
AbsPoint P 14 1 / 3 d 5 ;
```

If the model gets scaled by, say, 10, only the constant parameters of P will be scaled, hence P will become:

```
AbsPoint P 14 1 / 30 d 50 ;
```

Since x will also be dilated, the coordinates of P will be 30, 90, 50 and everything will be fine. However, in case of non-uniform scaling, it is all wrong: variables are always dilated by the average scale factor $(\alpha_x \alpha_y \alpha_z)^{1/3}$, therefore P will end up dilated by α_x, $(\alpha_x \alpha_y \alpha_z)^{1/3}$, α_z instead of α_x, α_y, α_z.

Physically realistic formulas (i.e. without additive unitless constants) will be automatically properly scaled, when their geometric supports get scaled. Unrealistic formula will be defeated since they will contain unscaled constants

```
Variable d          / 9 L  ;
Formula  unscalable / { d + 1 } L  ;
```

Scale by 10 and d becomes 90, unscalable will be 91 instead of 100.

Rotation and translation of objects depending on variables

Variables do not get translated or rotated, therefore

```
AbsPoint P 14 1 / 3 d 5 ;
```

is untransformable, since its y coordinate cannot change. In conclusion:

Variable used as world coordinates will defeat non-uniform scaling, translation and rotation..

How to create models well behaved with respects to transformations?

Use physically realistic formulas.

Systematically create points relative to a frame, which will ensure proper rotation and translation.

Use variables with caution if you plan to scale non-uniformly.

See also: Formula

Wireframe

Parents and characteristic properties

Frame / Point = frame or point entity. insert location
Filename = name of .3DA or .PAT file

Description

A WireFrame entity is a .3DA or .PAT file which can be included in a model primarily for display or comparison purposes. It is not an entity which can be used in the construction of other SurfaceWorks entities. For example, a curve represented in a WireFrame entity cannot support a bead, and it cannot be used as an edge of a surface.

Filename is the filename of a .3DA or .PAT file. If no extension is included, SurfaceWorks assumes .3DA.

A .3DA or .PAT file is an ASCII file consisting of an arbitrary number of lines, each terminated (in MS-DOS) by a carriage-return-line-feed combination. Each line has 4 numerical entries: pen, x, y, z. 'pen' is the number of a pen color for drawing to the point (x,y,z); if 'pen' is 0, the pen is up in moving to (x,y,z).

Here's an example of a .3DA file for a cube:

```
 0  -1  -1  -1
13   1  -1  -1
13   1   1  -1
13  -1   1  -1
13  -1  -1  -1
 0  -1  -1   1
13   1  -1   1
13   1   1   1
13  -1   1   1
13  -1  -1   1
 0  -1  -1  -1
13  -1  -1   1
 0   1  -1  -1
13   1  -1   1
 0   1   1  -1
13   1   1   1
 0  -1   1  -1
13  -1   1   1
```

Frame/Point is the name of a frame entity. This specifies an insert origin and orientation for the WireFrame entity, allowing the WireFrame to be placed in any position you desire.

When Frame/Point is a frame, the wireframe file coordinates are interpreted as frame coordinates x,y,z.

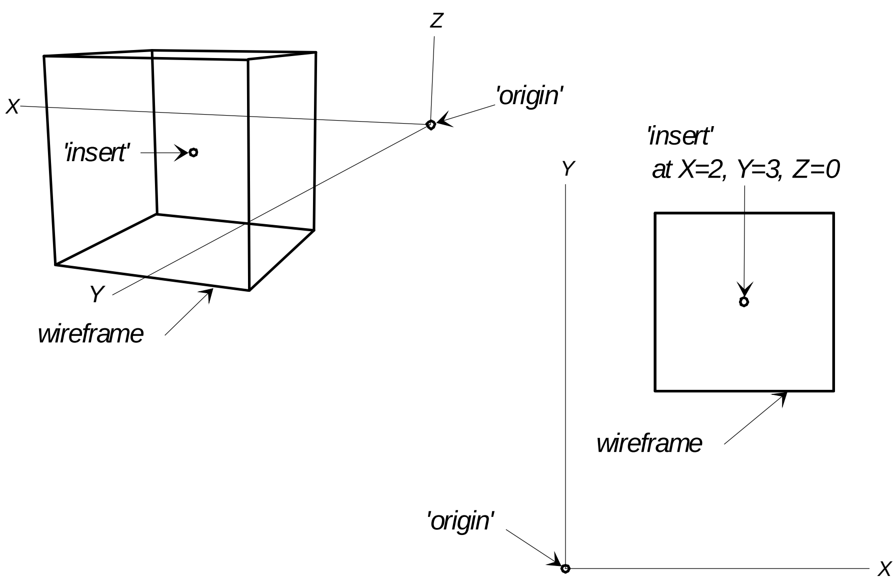
When Frame/Point is a point, the point's coordinates are added to the wireframe coordinates as the file is read in (equivalent to a frame located at point and parallel to the world coordinates).

When Frame/Point is the default frame '*', the wireframe file coordinates are interpreted as world coordinates X,Y,Z.

Color: Check the Use 3DA colors checkbox if you want to use the colors that are called out in the .3DA file. Uncheck the Use 3DA colors, to use any of the standard SurfaceWorks colors.

Visibility will represent the WireFrame entity with wires, points, or both (note that these points are not SurfaceWorks entities in and of themselves and, therefore, cannot be edited in SurfaceWorks).

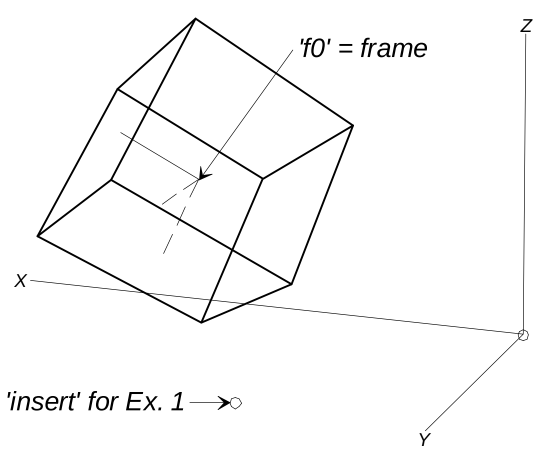
Example: wirefrm1



wirefrm.

(This example requires cube1.3da). 'cube' is a visible wireframe entity made from the file cube.3da. 'insert' is the point where the .3DA file's origin of coordinates will be placed. This means that 2 units will be added to each of the X-values and 3 units will be added to each of the Y-values in cube1.3da before the cube is displayed (and zero will be added to each of the Z-values).

Example: wirefrm2



wirefrm2.

(This example requires cube1.3da). This example shows the same cube1.3da wireframe as in the previous example, but this time it is inserted with reference to the Frame entity ‘f0’ — this allows the cube to be rotated with respect to the world coordinates.

X-spline Curve

Parents and characteristic properties

point1, point2 ... pointN = control points
type = direction of orientation of X-spline Curve
ecc = end condition code for the two directions other than the direction of orientation (specified by type)
s/m1, s/m2, s/m3, s/m4 = slope or moment at each end of the curve for the two directions other than the direction of orientation (specified by type)

Description

The X-spline Curve was created to add 100% forward compatibility from FL/2B to SurfaceWorks. The X-spline Curve, “explicit spline” or X-spline, is a cubic spline passing through its data points, with knots at the interior data points. In this respect, it is like a type-3 C-spline Curve. However, there are several important differences from a C-spline Curve:

1 The X-spline Curve is an explicit function of either X, Y, or Z, according to its type. Its natural or default t parameter is distributed in proportion to distance along the specified axis.

2 A consequence of the above is that an X-spline Curve must be oriented with respect to its selected axis, and must have that coordinate either strictly increasing or strictly decreasing along its length; it isn’t allowed to double back on itself.

3 If you rotate the set of points that parent an X-spline Curve about any line not parallel to the selected axis, the curve changes shape, as well as position. If you rotate the points far enough, the X-spline Curve will cause a geometry error.

4 At each end of an X-spline Curve, you have explicit control over either the slope or the curvature (bending moment) in two directions. To be more specific, suppose we have a type-1 XCurve, i.e., oriented with respect to the X-axis. At each end of this curve, you will explicitly specify either the slopes Y’, Z’ or the moments Y”, Z”.

5 An X-spline Curve will not in general lie in a plane, even if all its control points do. There are somewhat intricate constraints on the end condition values that are required to make the curve lie in a plane.

6 The Y and Z projections of a type-1 X-spline Curve are completely independent. That is, the Y-projection depends only on the Z-coordinates and the Z’ or Z” end condition values, and vice versa.

type specifies the direction of orientation of the X-spline Curve:

1 for X orientation

2 for Y orientation

3 for Z orientation

Note: this usage of type has nothing to do with "spline" type as used in B-spline Curve, C-spline Curve entity specifications. X-spline Curves are always cubic splines.

---

ecc is the end condition code for the two directions other than the direction of orientation (specified by type):

0 = moment both ends

1 = slope at t=0 end, moment at t=1 end

2 = moment at t=0 end, slope at t=1 end

3 = slope at both ends

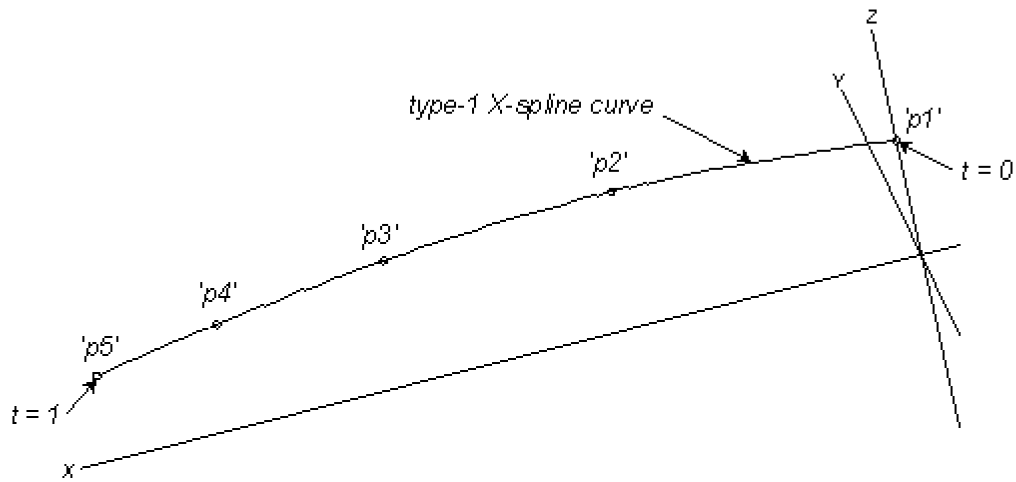
s/m1, s/m2, s/m3, s/m4 are the slope or moment at each end of the curve for the two directions other than the direction of orientation (specified by type):

	type-1 (X orientation)	**type-2 (Y orientation)**	**type-3 (Z orientation)**	
s/m1	Y' or Y"	Z' or Z"	X' or X"	at t = 0 end
s/m2	Z' or Z"	X' or X"	Y' or Y"	at t = 0 end
s/m3	Y' or Y"	Z' or Z"	X' or X"	at t = 1 end
s/m4	Z' or Z"	X' or X"	Y' or Y"	at t = 1 end

While inclusion of the X-spline Curve was motivated primarily by the desire for full compatibility with FAIRLINE/2, it certainly has some other worthwhile uses. For example... boat hull with straight frames... cambered deck as ALoftSurf... accurately flat spray chines...

Example xcurve.ms2

This example is a candidate sheerline for an 8-meter sailing yacht. You'll probably want to use Edit/ Model File to follow the discussion. The X-spline Curve is type-1 (oriented to the x-axis), and has end condition code = 1 (slope at t=0 end, moment at t=1 end). It passes through the five points in sequence. At t=0, the Y-slope (dY/dX) is 0.5, and the Z-slope (dZ/dX) is -0.13. At t=1, the Y-moment (d^2Y/dX^2) is -.02, and the Z-moment (d^2Z/dX^2) is .005.



XCURVE.MS2 at Lat -30, Lon 60.

See also: X-spline Lofted Surface

X-Spline Lofted Surface

Parents and characteristic properties

curve1, curve2 ... curveN = master curves (t-orientation doesn't matter)
type = direction of orientation of XLoftSurf
ecc = end condition code for the two directions other than the direction of orientation (specified by type)
graph1, graph2, graph3, graph4 = names of BGraph objects or '*' for default graph = end condition data for the two directions other than the direction of orientation (specified by type)

Description

The X-spline Lofted Surface, like other lofted surfaces, is parented by two or more master curves. For the X-spline Lofted Surface, the number of master curves is arbitrary. The XLoftSurf is lofted with X-splines. Like the X-spline Curve, the X-spline Lofted Surface needs end condition data, which is supplied by the four graphs.

type specifies the direction of orientation of the lofting X-splines:

1 for X orientation

2 for Y orientation

3 for Z orientation

Note: this usage of type has NOTHING to do with "spline" type as used in B-spline Lofted Surface, C-spline Lofted Surface, etc. entity specifications. X-spline Loftec Surfaces use cubic splines.

ecc is the end condition code for the lofting X-splines, for the two directions other than the direction of orientation (specified by type):

0 = moment at both edges

1 = slope at v=0 edge, moment at v=1 edge

2 = moment at v=0 edge, slope at v=1 edge

3 = slope at both edges

graph is the name of a B-spline Graph object. The four B-spline Graphs supply the end condition data for the lofting X-splines, for the two directions other than the direction of orientation (specified by type)

	type-1 (X orientation)	type-2 (Y orientation)	type-3 (Z orientation)	
graph1	Y’ or Y”	Z’ or Z”	X’ or X”	at first MC
graph2	Z’ or Z”	X’ or X”	Y’ or Y”	at first MC
graph3	Y’ or Y”	Z’ or Z”	X’ or X”	at last MC
graph4	Z’ or Z”	X’ or X”	Y’ or Y”	at last MC

relabel is used to control the labeling of the v-direction (the lofting curves) of the surface. See “Relabeling Curves and Snakes” in the Editing Models chapter.

The type-1 X-spline Lofted Surface can reproduce exactly any FAIRLINE/2 surface. In this case, the B-spline Graphs must be the same type as the end master curves and must have one component for each master curve vertex. The slope or moment columns in the FAIRLINE/2 Rep can be copied directly into the B-spline Graphs. To automatically perform the translation of a FAIRLINE/2 Rep file into Points, B-spline Curves, B-spline Graphs, and an X-spline Lofted Surface, you can use the utility program FL2MSF (see Appendix B).

Crossed Master Curves

Users of X-spline Lofted Surfaces should be aware that they are vulnerable to a special error condition that can cause unexpected and puzzling results: the dreaded "Crossed Master Curves" error (Error 247 or 248). To locate a point at parameters u,v on an X-spline Lofted Surface, SurfaceWorks first evaluates each master curve (and each end condition graph) at t = u, then it passes an X-spline Curve through the points in order, then it evaluates the X-spline Curve at t = v. The problem arises in the second stage if the points obtained from the master curves don't form a strictly ascending or descending sequence in the coordinate corresponding to type, e.g. the X-coordinate for type-1. The construction of the X-spline Curve fails, and the surface is marked as in error; any descendants of the XLoftSurf get error 284.

This error can occur even when the master curves don't actually cross within the nominal u-range of 0 to 1. One way it happens is a consequence of SurfaceWorks extending the tabulation of curves and surfaces for one subdivision beyond 0 and 1. Suppose you have specified 10 divisions and 1 subdivision in the u-direction; then SurfaceWorks actually tabulates the surface from u = -0.1 to u = 1.1. If master curves are approaching each other near the 0 (or 1) end of the range, their extensions can actually cross a little ways beyond 0 (or 1), and this "invisible" crossing will trigger error 248. Two workarounds are possible:

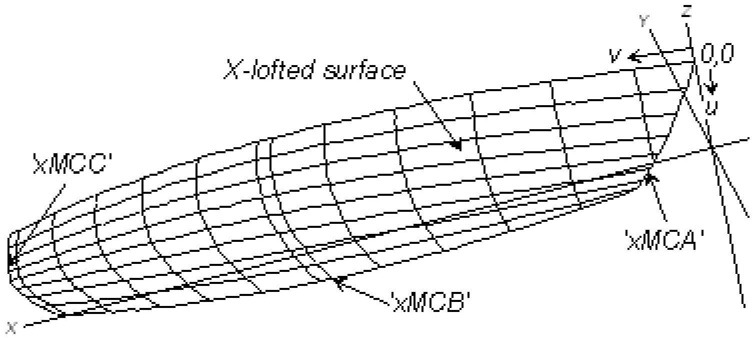
1. try increasing the subdivisions for the u-direction, since a smaller subdivision means the tabulated surface extends a shorter distance beyond u = 0 (or 1)

2. change the master curves so they are farther apart at u = 0 (or 1)

Another way this error can be triggered is by dragging a magnet near or beyond the u = 0 or u = 1 edge of an X-spline Lofted Surface, where master curves are converging. As the program performs its search for the u,v coordinates on the surface corresponding to the visual position of the cursor, the surface might be evaluated at points outside the range u = 0 to u = 1. If the error occurs while dragging a magnet, you can usually clear it by choosing Edit>Model File, then just <Ok> without making any changes.

Likewise, the error can occur during the solution for position of a ProjMagnet, IntMagnet, ProjSnake, or IntSnake on an X-spline Lofted Surface. It is especially important on an X-spline Lofted Surface to use a magnet to specify the starting position of one of these objects. This helps a great deal to limit the area that the program has to search in, reducing the chance that it will look beyond the u = 0 or u = 1 edges.

Example defaultx.ms2



DEFAULTX.MS2 at Lat -30, Lon 60.

DEFAULTX.MS2 is the FL/2B Default Rep converted to a SurfaceWorks model file by the utility program FL2MSF (for FL2MSF details, see Appendix B). The SurfaceWorks model is formed by three type-2 B-spline master curves and an X-spline Lofted Surface. You'll probably want to use Edit>Model File to follow the discussion. There are four B-spline Graphs which distribute bending moment along the end master curves of the X-spline Lofted Surface. The B-spline Graphs are type-2, the same type as the B-spline master curves, and they have the same number of values as the number of vertices in each of the master curves — 3 for xMCA and 4 for xMCC.

See also: Xcurve

XYZ Bead

Parents and characteristic properties

Bead/Curve = curve or bead. designates the parent curve
Location = specification of X, Y, or Z constraint and the value for it
Relative to (when checked) = the parent point for relative coordinates

Description

An XYZ Bead is a point constrained to lie on a curve, but located by Cartesian coordinates rather than by t parameter value. To make an XYZ Bead, SurfaceWorks makes an automated search for the location on the parent curve that lies at the specified X, Y, or Z (or dX, dY, or dZ offset from another point).

If you get an odd result with an XYZ Bead, try putting a bead on the curve near where you want the XYZ Bead to go and using the bead as the parent.

Bead/Curve can be a bead (or ring) or a curve (or snake):

When Bead/Curve is a bead, it serves three purposes:

1. It specifies which curve the XYZ Bead is to lie on: the curve that is the parent of the bead.
2. It specifies the starting point for SurfaceWorks to begin its search, thus helping it find the intersection faster and more reliably.
3. In case of multiple intersections of the curve with the specified X, Y, or Z constraint, the XYZ Bead will be located at the intersection nearest to the bead.

When Bead/Curve is a curve, it directly specifies the parent curve for the XYZ Bead. The search will begin at the default starting point of t = 0.

Note: If you use a bead for Bead/Curve, place it close to the intersection you want — the closer, the better (within reason — you want SurfaceWorks to calculate the intersection, so don't go to the trouble of calculating it yourself!)

Location for an XYZ Bead has two components: identification of which Cartesian coordinate (X, Y, or Z) you wish to specify and the value for that coordinate. When there are multiple intersections of the curve with the specified X, Y, or Z constraint, SurfaceWorks uses the one closest to:

- The t = 0 end of the curve (when Bead/Curve is a curve)
  or
- The bead used to identify the curve (when Bead/Curve is a bead)

If you want one of multiple solutions that is not the one closest to the t = 0 end of the curve, use a bead for Bead/Curve and put it close to the intersection you want.

If you want to put an XYZ Bead on the extension of its parent curve, we recommend that you use a bead for Bead/Curve; put it off the end of the curve, relatively near where the XYZ Bead will be located.

Note: If a curve lies entirely in a plane parallel to the X, Y, or Z axis, the Cartesian coordinate in the direction normal to that plane will be grayed out; e.g. if the curve lies in a plane parallel to the X-axis, the X choice for Location will be grayed out. (Why? Imagine a curve that lies in the X=4 plane; all locations along the curve have an X-value of 4. Locating the bead by X-value would mean the bead could be anywhere along the curve.)

Relative to: If you want to make the coordinates for the XYZ Bead relative to (offset from) another point, check the box and specify the parent point. The parent point for an XYZ Bead can be any kind of point entity (3D point, bead, magnet, ring), located anywhere in the model. Note that this is different from a (parametric) Bead for which the “Relative to” parent can only be a bead entity and the parent must lie on the same curve as the Bead.

When you make a relative XYZ Bead, you must select both a Bead/Curve and a “Relative to” entity. This is because “Relative to” can be any kind of point entity and, therefore, does not automatically identify the curve the XYZ Bead is to lie on.

Changing properties and parents

If you change:

- Your coordinate choice (e.g. from X to Y)

SurfaceWorks automatically will adjust the location value:

- When there is a single solution, SurfaceWorks will maintain the XYZ Bead’s position.
- When there are multiple solutions, SurfaceWorks will use the solution closest to the t = 0 end of the curve or, if a bead is used for Bead/Curve, the solution closest to the bead.

If you change:

- The parent point (when “Relative to” is checked)
- Between absolute / relative coordinates (toggle the check for “Relative to”)

SurfaceWorks asks if you want to "Preserve Absolute Location" or Preserve Offset Values".

Example: XYZBead1

This model demonstrates XYZ Beads defined in several different ways, including choosing from multiple possible solutions.

'e0' is constrained to lie at X=5.93 on curve 'A' and 'e5' is similarly constrained to lie at Z=1 on line 'l2'.

'e4' lies on Line 'l0' at 2.471 X-units from its Relative to Bead, 'e2'

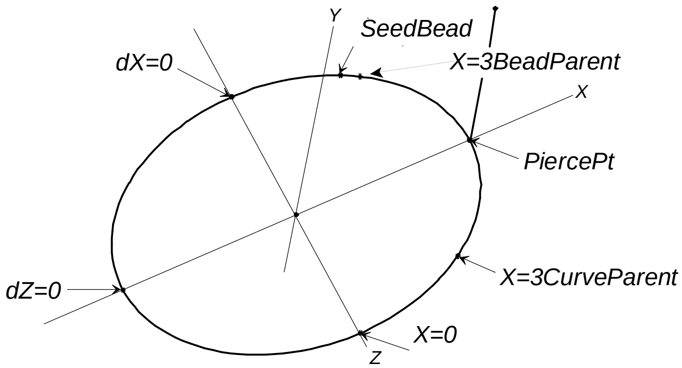
'e1' has the same X-coordinate as Bead 'e2', its Relative to point, and lies on curve 'A', its Bead / Curve.

'e3' lies on curve 'A', but is offset from its Relative to Bead, 'e4' on LIne 'l0'

Dragging 'e1', 'e3' or 'e5' shifts the endpoints of the Lines 'l2' and 'l3'.

Dragging 'e4' changes the location of 'l3's endpoint indirectly.

Dragging 'e2' shifts the whole collection of lines.



XYZBead1

See also: Bead, Ring, Arc-length Bead, Arc-length Ring, XYZ Ring

XYZ Magnet

Characteristic Data

Located by = how located
Magnet/surface = destination surface for the magnet
Frame = frame for Located by
Point = optionalreference point
Location1, Location 2 =

Description

Located by determines whether you are making the YZ, ZX, or XY variety:

Y and Z positions or offsets locates the magnet by its Y and Z locations

Z and X positions or offsets locates the magnet by its Z and X locations

X and Y positions or offsets locates the magnet by its X and Y locations.

Note these designations are based on an "odd-man-out" rule:

Type-1 is the one that doesn't use X

Type-2 is the one that doesn't use Y

Type-3 is the one that doesn't use Z

Magnet/surface designates the host surface for the XYZMagnet. If a magnet is used here, it also provides a starting location for the iterative solution. Unless there is clearly only one solution (considering also the extension of the surface beyond the 0 to 1 nominal parameter range), it is always preferable to specify a magnet that is somewhere in the neighborhood of the intended intersection. If you just specify the surface, the u = 0.5, v = 0.5 position on the surface will be used as a starting value for the XYZMagnet. If the true intersection is far from 0.5, 0.5, the XYZMagnet may fail to converge, or may go to an unexpected location on the surface extension.

Frame specifies the coordinate frame in which to measure X, Y or Z. The default frame '*' is the global coordinate system, in which case the coordinates used are just the global X, Y or Z. If you choose to measure offsets or location in another Frame not aligned with the global coordinates, Location1 and Location2 will be frame coordinates, x, y or z.

Point is an optional reference point. If you choose the default point '*', the Locations will be measured from the origin of Frame. This creates the "absolute" variety of XYZMagnet. If you choose another point for Point, the Locations will be relative to that Point. This makes the "relative" variety of XYZMagnet.

Location1, Location2 are the two coordinate offsets.

Example

Suppose you need to locate a point 'm1' on the deck surface, exactly 3.20 ft. from centerplane and 10.46 ft. aft of the forward reference station. The most direct way to do that now is an XYZMagnet. Since X and Y are specified, the type has to be 3. Use '*' for both Frame and Point (coordinates measured in the global coordinate system), and 10.46 and 3.20 for the Locations.

Suppose you need another point 'm2' on the deck, exactly 2.25 ft. aft of 'm1' and the same distance from the centerplane. The simplest way to do this is an XYZMagnet, again type-3 since X and Y are specified, with Frame = '*', Point = 'm1', and Locations of 2.25 and 0.

Sample file:

XYZ MAGNET.MS2

XYZ Ring

Parents and characteristic properties

Ring/Snake = snake or ring. designates the parent snake
Location = specification of X, Y, or Z constraint and the value for it
Relative to (when checked) = the parent point for relative coordinates

Description

An XYZ Ring is a point constrained to lie on a snake, but located by Cartesian coordinates rather than by t parameter value. To make an XYZ Ring, SurfaceWorks makes an automated search for the location on the parent snake that lies at the specified X, Y, or Z (or dX, dY, or dZ offset from another point).

If you get an odd result with an XYZ Ring, try putting a ring on the curve near where you want the XYZ Ring to go and using the ring as the parent.

Ring/Snake can be a ring or a snake:

When Snake is a ring, it serves three purposes:

1. It specifies which snake the XYZ Ring is to lie on: the snake that is the parent of the ring.
2. It specifies the starting point for SurfaceWorks to begin its search, thus helping it find the intersection faster and more reliably.
3. In case of multiple intersections of the snake with the specified X, Y, or Z constraint, the XYZ Ring will be located at the intersection nearest to the ring.

When Ring/Snake is a snake, it directly specifies the parent curve for the XYZ Ring. The search will begin at the default starting point of t = 0.

Note: If you use a ring for Ring/Snake, place it close to the intersection you want — the closer, the better (within reason — you want SurfaceWorks to calculate the intersection, so don't go to the trouble of calculating it yourself!)

Location for an XYZ Ring has two components: identification of which Cartesian coordinate (X, Y, or Z) you wish to specify and the value for that coordinate. When there are multiple intersections of the snake with the specified X, Y, or Z constraint, SurfaceWorks uses the one closest to:

- The t = 0 end of the curve (when Ring/Snake is a snake)
  or
- The ring used to identify the snake (when Ring/Snake is a ring)

If you want one of multiple solutions that is not the one closest to the t = 0 end of the snake, use a ring for Ring/Snake and put it close to the intersection you want.

If you want to put an XYZ Ring on the extension of its parent snake, we recommend that you use a ring for Ring/Snake; put it off the end of the snake, relatively near where the XYZ Ring will be located.

Note: If a snake lies entirely in a plane parallel to the X, Y, or Z axis, the Cartesian coordinate in the direction normal to that plane will be grayed out; e.g. if the snake lies in a plane parallel to the X-axis, the X choice for Location will be grayed out. (Why? Imagine a snake that lies in the X=4 plane; all locations along the snake have an X-value of 4. Locating the ring by X-value would mean the ring could be anywhere along the snake.)

Relative to: If you want to make the coordinates for the XYZ Ring relative to (offset from) another point, check the box and specify the parent point. The parent point for an XYZ Ring can be any kind of point entity (3D point, bead, magnet, ring), located anywhere in the model. Note that this is different from a (parametric) Ring for which the "Relative to" parent can only be a ring or a bead entity and the parent must lie on the same snake as the Ring.

When you make a relative XYZ Ring, you must select both a Ring/Snake and a "Relative to" entity. This is because "Relative to" can be any kind of point entity and, therefore, does not automatically identify the snake the XYZ Ring is to lie on.

Changing properties and parents

- Your coordinate choice (e.g. from X to Y)

SurfaceWorks automatically will adjust the location value:

- When there is a single solution, SurfaceWorks will maintain the XYZ Ring's position.
- When there are multiple solutions, SurfaceWorks will use the solution closest to the t = 0 end of the snake or, if a ring is used for Ring/Snake, the solution closest to the ring.

If you change:

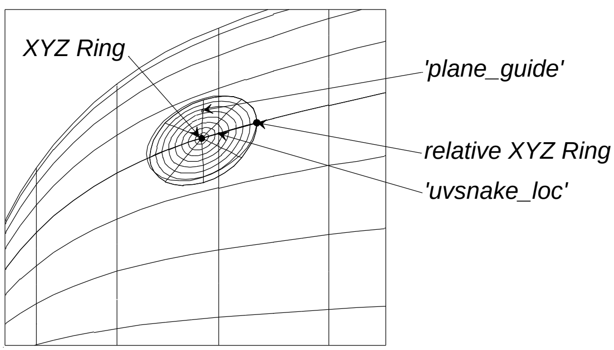
- The parent point (when “Relative to” is checked)
- Between absolute / relative coordinates (toggle the check for “Relative to”)

SurfaceWorks asks if you want to "Preserve Absolute Location" or "Preserve Offset Values".

Example: XYZRings1

This model uses an XYZ Ring to position the center of the round patch at X = 2.

‘surface1’ (dark cyan) is a Revolution Surface. ‘patch’ (green) is a SubSurface on ‘surface1’. ‘snake1’ (magenta) is a v=constant UVSnake parented by the Magnet (red) ‘uvsnake_loc’. Two XYZ Rings (both blue) lie on the UVSnake: ‘XYZRing’ and ‘relXYZRing’ are the center and start points, respectively, for the Arc ‘circle’; the Magnet ‘plane_guide’ is its plane guide. ‘boundary’ (magenta) is the projection of ‘circle’ onto ‘surface1’. ‘boundary’ and ‘XYZRing’ are the bounding curves for the SubSurf ‘patch’.



XYZRings1

The point entities are handles you can use to modify the model. For instance, you can:

Drag ‘XYZRing’ to different X-locations along ‘snake1’.
Drag ‘relXYZRing’ to change the size of ‘patch’.
Drag ‘plane_guide’ to modify the circular / elliptical shape of ‘patch’ (‘patch’ is a true circle in u-v space).
Drag ‘uvsnake_loc’ to change the vertical (in this view) location of ‘patch’.

See also: Ring, Bead, Arc-length Bead, Arc-length Ring, XYZ Bead

Entities Listed by Class

Lists all entities available. Each entry has a link to its description in the Entity Descriptions book.

Point Entities

3D Points	Beads	Magnets	Rings
Point	Bead	Magnet	Ring
Blended Point			
Center Point			
Copy Point	Copy Bead	Copy Magnet	Copy Ring
Intersection Point	Intersection Bead	Intersection Magnet	Intersection Ring
Mirrored Point			
Offset Point			
Projected Point		Projected Magnet	
Rotated Point			
Tabulated Point			
Tangent Point		Tangent Magnet	
	Arc-length Bead		Arc-length Ring
	Proximity Bead	Proximity Magnet	Proximity Ring
	XYZ Bead	XYZ Magnet	XYZ Ring

Curve and Snake Entities

Curve entities	Snake entities
Arc	Arc Snake
B-spline FittedCurve	B-spline Fitted Snake
B-spline Curve	B-spline Snake
C-spline Curve	C-spline Snake

Conic Section	
Contour Curve	
Copy Curve	Copy Snake
	Edge Snake
Expanded Curve	
Foil Curve	Foil Snake
Helix	
	Geodesic Snake
	Intersection Snake
Line	Line Snake
Mirrored Curve	
NUB-spline Fitted Curve	NUB-spline Fitted Snake
NURBS Curve	NURBS Snake
Offset Curve	
PolyCurve	PolySnake
Procedural Curve	Procedural Snake
Projected Curve	Projected Snake
Radius Arc	
Relative Curve	Relative Snake
Rotated Curve	
SubCurve	SubSnake
Tabulated Curve	
X-Spline Curve	
	UVSnake

Surface Entities

Arc Lofted Surface
Blend Surface
Blister
B-spline Fitted Surface
B-spline Lofted Surface
B-spline Surface
Centerpoint Boundary Surface
Copy Surface
C-spline Lofted Surface
Developable Surface
Expanded Surface
Foil Lofted Surface
Mirrored Surface
NUB-spline Fitted Surface
NURBS Surface
Offset Surface
PolySurface
Procedural Surface
Projected Surface
Relative Surface
Revolution Surface

Parametric Solids

Triangle Mesh Entities

Other Entities

Macros

Introduction

Although it is certainly not necessary to know how to program macros in order to use SurfaceWorks, the information contained in this document provides the advanced user with very powerful capabilities. Macros can enable you to:

- Add custom features to SurfaceWorks
- Automate frequently repeated tasks and calculations
- Utilize SurfaceWorks as a powerful “geometry engine” within a system of Windows programs

In this document we assume that you know something about programming. If you are new to programming with Visual Basic, we recommend that you read the *Programmer’s Guide* that comes with Visual Basic.

What is OLE Automation?

OLE Automation is a technology by which a program written in Visual Basic, Visual Basic for Applications (VBA) or C/C++ can utilize objects exposed by other applications such as SurfaceWorks The chart below diagrams the current SurfaceWorks OLE Automation objects and their relationships.

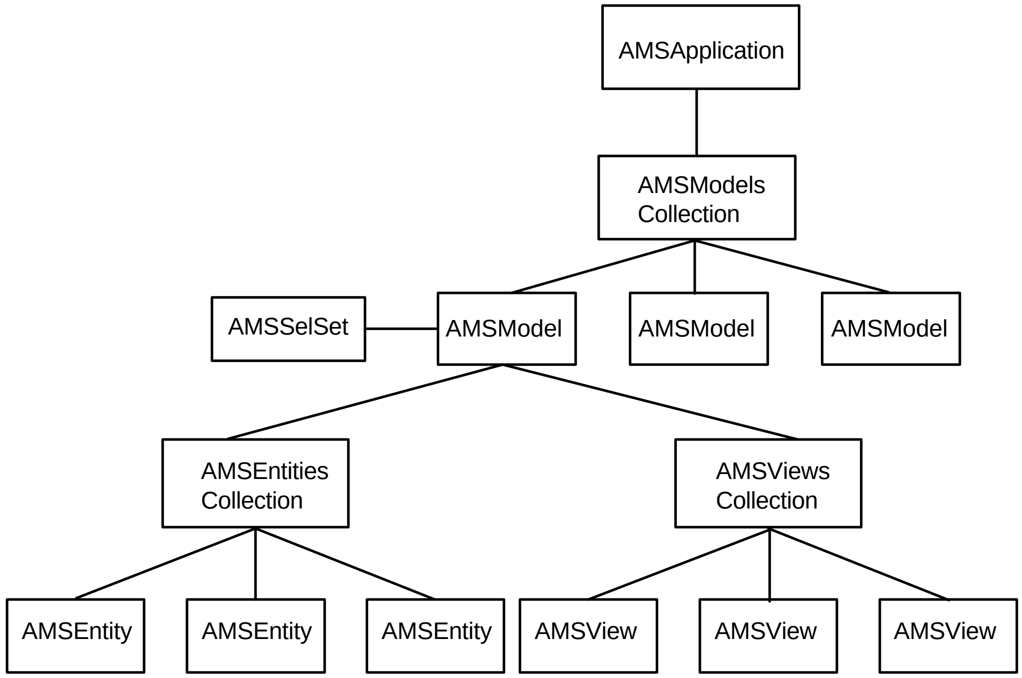


Chart: SurfaceWorks OLE Automation objects and their relationships.

Each of these objects has its own set of properties and methods that can be used in a macro. Methods are actions that an object can perform while properties read and set information about the state of an object.

Applications that expose objects through OLE Automation such as SurfaceWorks are known as OLE automation servers. Applications that use these objects such as any macros that you write for SurfaceWorks are known as OLE Automation clients or OLE Automation controllers.

Of the different programming languages that can be used, Microsoft's Visual Basic is probably the easiest language to use to create macros. Therefore, this document is limited to describing the creation of macros only with the use of Visual Basic, however, the principles outlined here are useful when using other programming languages.

SurfaceWorks Objects and Their Properties and Methods

There are 8 objects accessible in SurfaceWorks, each of which has a set of Properties and Methods that apply to it.

AMSApplication

This is the application itself. This object allows you to access application level information. The AMSApplication is required to access the AMSModels collection.

AMSModel

This is a model. With the AMSModel object, you can create and edit entities. The AMSModel object allows access to the AMSEntities collection.

AMSModels

This is the collection of models in the AMSApplication. This collection gets you access to a particular model if more than one is open, and allows you to add and close a model.

AMSEntity

This is a particular entity. This object gives you access to the details of each entity.

AMSEntities

This is the collection of SurfaceWorks entities. This object gives you access to all of the available entities.

AMSSelSet

This is the Selection Set. This object gives you access to the Selection Set.

AMSView

This is a particular view. This object has only read-only Properties, except for the Close Method.

AMSViews

This is the collection of possible views. This object allows you to set and read a number of view Properties.

When used to in a program, these require the prefix **SWAdvMod.** as in **SWAdvMod.AMSSelSet**. SWAdvMod stands for SurfaceWorks Advanced Modeler.

Getting the SurfaceWorks Application Object

Before you can do anything with SurfaceWorks from a macro, you need to obtain SurfaceWorks Application object.

Use the GetObject function to access a model in a current session of SurfaceWorks. Here's an example of its use:

```
Dim AppObj As SWAdvMod.AMSApplication
Set AppObj = GetObject(, "SWAdvMod.Application")
```

Once you have the Application object, it is easy to gain access to the other objects because each object has properties that return values that refer to other objects. For example, the Models property of the Application object refers to the AMSModels collection, which, in turn, has a Model property through which you can access any open model. In code it would look like this:

```
Dim swModels As SWAdvMod.AMSModels
Dim swModel As SWAdvMod.AMSModel
Set swModels = AppObj.Models
Set swModel = swModels.Model(1)
```

In addition to being able to travel down the chain of hierarchy in this manner, it is also possible to access higher level objects from most of the objects. For example, every AMSEntity object has a Model property that enables you to access the model to which it belongs.

Collection Objects

A collection is a group of objects of the same type. Though similar to an array, an object's position in a collection does not remain fixed but can be changed when another object is added or deleted. You can access an individual object in a collection or iterate through a collection to perform the same set of operations on all the objects in that collection.

To access an individual model in the AMSModels collection, several alternatives are provided. For example, if you have loaded several models in SurfaceWorks and the first one has the file name “boat.ms2”, either of the following lines of code would give you access to that model:

```
Set swModel = swModels.Model(1)
Set swModel = swModels.Model(“boat”)
```

Since Model is the default method / property of the AMSModels collection, it is actually not required to include Model in these lines of code, so you could also do either of these:

```
Set swModel = swModels(1)
Set swModel = swModels(“boat”)
```

Likewise, if you had a point named “Pt1” which was the first object in a model, any of the following lines of code would gain access to it from the AMSEntities collection:

```
Set entity = entities.Entity(1)
Set entity = entities.Entity(“Pt1”)
Set entity = entities(1)
```

```
Set entity = entities("Pt1")
```

You can iterate through the AMSModels or AMSEntities collections using a For…Next loop and the Count property. For example, let's say you want to put each model's file name in an array to be used later for some purpose. You could do this with the following code:

```
Set swModels = AppObj.Models
For i = 1 To swModels.Count
    Set swModel = swModels(i)
    FileNames(i) = swModel.Name
Next i
```

Another way to iterate through collections is to use the Visual Basic For Each…Next statement which can be used by both the AMSModels and AMSEntities collections. One of the main differences between this method and using the For…Next loop is that you can perform a set of operations on all items in a collection without needing to know the total number of items in the collection. So if you wanted to move all Offset Points (OffsetPts) by 10 units in the x-direction, you could do it this way:

```
Dim entity As SWAdvMod.AMSEntity
Dim entities As SWAdvMod.AMSEntities
Dim PtFloats As Variant

Set entities = swModel.Entities
For Each entity In entities
    If entity.Type = "OffsetPt" Then
        PtFloats = entity.Floats
        PtFloats(1) = PtFloats(1) + 10
        entity.Floats = PtFloats
    End If
Next entity
```

Using Properties and Methods

All properties return a value that may be a reference to some other object or a string or numeric data type. If the property returns an object reference, the variable that receives that reference must be declared as type Object or as the specific object being referenced. When assigning a value to a variable of type Object, remember to use the Set statement.

```
Dim entity1 As Object
Set entity1 = entities(1)

Dim entity2 As SWAdvMod.AMSEntity
Set entity2 = entities(2)
```

The latter approach is favored since it results in faster performance and any run-time errors will be reported faster.

Some properties allow you to get or set their values, while other properties only allow you to get their values. A property whose value you can only get is said to be "read-only." A few of the properties take arguments. Here are some examples of getting and setting properties:

```
Set swModel = AppObj.ActiveModel
dist = entity1.Clearance(entity2)
entity.Name = "Pt1"
SelObjs = swSelSet.SelectedEntities
```

Many of the methods correspond to actions that can be performed in SurfaceWorks. A few of the methods return a value. For example, the Open method returns the AMSModel object that is created when the file is opened.

Helpful Facts and Suggestions

SurfaceWorks Entity Names: User vs. Internal

The names that are used in the user interface to SurfaceWorks (the ones you and I use) are different from the names used internally by SurfaceWorks. When you write a macro, you must use the internal names. A list of all SurfaceWorks entities and their internal names is found in Entities Syntax at the end of this chapter.

Disabling Interactive Input

For most macros, you will want to disable any input to SurfaceWorks while the macro is running. To do this set the **Interactive** property to **False** as soon as you obtain the AMSApplication object. But be sure to set **Interactive** back to **True** wherever the macro may exit or you will find yourself unable to activate SurfaceWorks again. If you find yourself stuck in this situation, you have two choices. The first is to shut SurfaceWorks down by opening Task Manager, selecting SurfaceWorks from the list and clicking on the **End Task** button and then restarting it. Unfortunately, this choice will result in the loss of any unsaved work that was in progress. A better way is to create another macro that sets the **Interactive** property to **True** and execute it. Then you will be able to proceed without the loss of any work.

If you have set **Interactive** to **False** and your macro uses a modal dialog, set **Interactive** to **True** right before bringing up the dialog and set it back to **False** right after returning from the dialog. If you do not do this and you have other applications running at the time, one of those applications will come to the top after closing the dialog and SurfaceWorks will no longer be the active application. This doesn't produce any adverse effect and you can easily activate and bring SurfaceWorks back to the top by clicking on its window or clicking its button on the **Task Bar** after the macro is finished, however it's a lot smoother if you employ the **Interactive** property.

Controlling Scope of Undo

You can set a sequence of operations performed on a model by a macro as an Undoable event by bracketing those operations with the **BeginUndoTransaction** and **EndUndoTransaction** methods of the **AMSModel** object. So if you want to be able to easily undo everything a macro has done, call the **BeginUndoTransaction** method before performing any changes to a model and then call **EndUndoTransaction** when all operations are completed. If you don't like what the macro did, just select Undo in SurfaceWorks and everything will be back to the way it was before the macro executed.

International Issues

If you are creating a macro that will be used in countries which use a comma as a decimal separator, beware of using the Visual Basic **CStr**, **CDbl**, **CSng**, **CInt** and **CLng** functions. All of these functions use the system locale to determine the decimal separator. However, SurfaceWorks always assumes that a period is the decimal separator, so use the **Str** and **Val** functions instead.

Property and Method Descriptions

AMSApplication

ActiveModel (Read-only Property)

Description:

Returns the AMSModel object that is currently active in SurfaceWorks.

Syntax:

object = appobj.ActiveModel

object - An expression that evaluates to an object.

appobj - An expression that evaluates to the AMSApplication object.

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

object = appobj.Application

object - An expression that evaluates to an object.

appobj - An expression that evaluates to the AMSApplication object.

Command (Method)

Description:

Causes SurfaceWorks to execute a command just as if it had been entered in the Command Window. To list the available command keywords, in SurfaceWorks pick **Tools> Command Window** and enter the command help. For a brief explanation of the action of any available command and its proper syntax, enter help command, where command is a command keyword. (For example, to get help on the CreateCopies command, enter help createcopies.) Some commands operate at the application level; most apply to the active model.

Syntax:

appobj.Command(command)

appobj - An expression that evaluates to the AMSApplication object.

command - A String expression that specifies a command string to be executed

ExportTolerance (Property)

Description:

Sets or returns the root mean square (RMS) tolerance used for approximating surfaces when exporting IGES files. The default value is calculated based on the model’s size.

Syntax:

appobj.ExportTolerance [= value]

appobj - An expression that evaluates to the AMSApplication object.

value - A numeric expression of type Double that specifies a new tolerance.

FullName (Read-only Property)

Description:

Returns a String holding the full path of the SurfaceWorks EXE.

Syntax:

appobj.FullName

appobj - An expression that evaluates to the AMSApplication object.

Interactive (Property)

Description:

Sets or returns a Boolean value indicating whether SurfaceWorks accepts actions from the user regardless of visibility.

Syntax:

appobj.Interactive [= bool]

appobj - An expression that evaluates to the AMSApplication object.

bool - A Boolean expression that evaluates to True or False.

Models (Read-only Property)

Description:

Returns the AMSModels object.

Syntax:

object = appobj.Models

object - An expression that evaluates to an object.

appobj - An expression that evaluates to the AMSApplication object.

Name (Read-only Property)

Description:

Returns a String holding the name of the SurfaceWorks application.

Syntax:

string = appobj.Name

string - An expression that evaluates to a string.

appobj - An expression that evaluates to the AMSApplication object.

Parent (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

object = appobj.Parent

object - An expression that evaluates to an object.

appobj - An expression that evaluates to the AMSApplication object.

PredefinedEntity (Read-only Property)

Description:

Returns one of SurfaceWorks' predefined entities.

Syntax:

object = appobj.PredefinedEntity(item)

object - An expression that evaluates to an AMSEntity object.

appobj - An expression that evaluates to the AMSApplication object.
item – Index of the desired predefined entity.

Index	Predefined Entity
1	*
2	*EMPTY*
3	*0
4	*1
5	*X=0
6	*Y=0
7	*Z=0
8	*UNIFORM

Quit (Method)

Description:

Terminates SurfaceWorks, closing all models.

Syntax:

appobj.Quit

appobj - An expression that evaluates to the Application object.

Version (Read-only Property)

Description:

Returns the version information of SurfaceWorks.

Syntax:

sVersion = appobj.Version

sVersion - A String expression.

appobj - An expression that evaluates to the Application object.

Visible (Property)

Description:

Sets or returns the visible state of SurfaceWorks.

Syntax:

appobj.Visible [= bool]

appobj - An expression that evaluates to the Application object.

bool - A Boolean expression that evaluates to True or False.

AMSModel

Activate (Method)

Description:

Activates the first window associated with the model.

Syntax:

modelobj.Activate

modelobj - An expression that evaluates to an AMSModel object.

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

object = modelobj.Application

object - An expression that evaluates to an object.

modelobj - An expression that evaluates to an AMSModel object.

BeginUndoTransaction (Method)

Description:

Begins recording any subsequent changes made to the model as an Undoable event. EndUndoTransaction ends the recording. Every call to BeginUndoTransaction must be followed by a call to EndUndoTransaction. Calls to BeginUndoTransaction cannot be nested, so EndUndoTransaction must be called before making another call to BeginUndoTransaction. The argument to this method is used in the description of the undo event given in SurfaceWorks' menu.

Syntax:

modelobj.BeginUndoTransaction sDescription

modelobj - An expression that evaluates to an AMSModel object.

sDescription - A String expression that is used to identfy the undo event in SurfaceWorks.

Close (Method)

Description:

Closes all windows associated with the model and removes the model from the AMSModels collection.

Syntax:

modelobj.Close savechanges

modelobj - An expression that evaluates to an AMSModel object.

savechanges – Can be any of the following values:
swaDoNotSaveChanges
swaPromptToSaveChanges
swaSaveChanges

Comment (Property)

Description:

Sets or returns a model's comment.

Syntax:

modelobj.Comment [= string]

modelobj - An expression that evaluates to an AMSModel object.

string - A String expression that specifies a new comment.

CurrentLayer (Property)

Description:

Sets or returns a model's current layer. When returning the current layer, it will return a Variant containing an array whose first element (index = 1) is the layer number and whose second element (index = 2) is the layer name. If the layer has no name the second element is an empty string.

Syntax:

modelobj.CurrentLayer [= layer]

modelobj - An expression that evaluates to an AMSModel object.

layer - A Variant expression that specifies the new current layer. layer can contain either a String identifying the layer name or an Integer identifying the layer number. If it is an Integer it must be between 0 and 255, inclusive.

DivMult (Property)

Description:

Sets or returns the model's divisions multiplier.

Syntax:

modelobj.DivMult [= dm]

modelobj - An expression that evaluates to an AMSModel object.

dm - An Integer expression whose value must be between 1 and 10, inclusive.

EndUndoTransaction (Method)

Description:

Ends the recording of all changes made to the model since the previous call to BeginUndoTransaction. See BeginUndoTransaction for a complete description of how these methods work.

Syntax:

modelobj.EndUndoTransaction

modelobj - An expression that evaluates to an AMSModel object.

Entities (Read-only Property)

Description:

Returns the AMSEntities collection.

Syntax:

modelobj.Entities

modelobj - An expression that evaluates to an AMSModel object.

Export3DAFile (Method)

Description:

Exports a 3DA file of the model. Symmetry images are included if the ExportSymmetric property has been set to True.

Syntax:

modelobj.Export3DAFile pathname, EntitiesToExport

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
EntitiesToExport - Can be either of the following values:
swaAllVisibleEntities
swaSelectionSet

Export3DDXFFile (Method)

Description:

Exports an AutoCAD DXF file of the model. Symmetry images are included if the ExportSymmetric property has been set to True.

Syntax:

modelobj.Export3DDXFFile pathname,LayerOption, SurfacesAsMeshes, ExportAsACADEntities, DXFVersion, EntitiesToExport

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
LayerOption – Can be any of the following values:
swaAllOnSingleLayer
swaModelLayers
swaAllOnIndividualLayers
swaAllAndEachContourCutOnIndividualLayers

SurfacesAsMeshes – A Boolean expression. When True, all surfaces will be exported as meshes. When False, surfaces are exported according to the value of ExportAsACADEntities.
ExportAsACADEntities - A Boolean expression. When True, all curves, isoparms, contours, arcs and lines will be exported as SPLINE, ARC and LINE entities. When False, these objects are exported as POLYLINE entities. Must be set to False if DXFVersion is set to swaR12_LT2.
DXFVersion – Identifies which dxf version the exported file will be saved as. Can be either of the following values:
swaR12_LT2
swaR2000_LT2000
EntitiesToExport - Can be either of the following values:
swaAllVisibleEntities
swaSelectionSet

ExportIGESFile (Method)

Description:

Exports an IGES file of the model. All exported surfaces are approximated as NURBS surfaces. Symmetry images are included if the ExportSymmetric property has been set to True.

Syntax:

modelobj.ExportIGESFile pathname, ExportedSurfs, AchievedTol, EntitiesToExport

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
ExportedSurfs – A Variant containing an array Objects. Each element is the AMSEntity object of a surface successfully exported.
AchievedTol – A Variant containing an array of Doubles. Each element in the array represents the tolerance achieved with the NURBS approximation of a surface. Each element corresponds with the surface having the same index in the ExportedSurfs array.
EntitiesToExport - Can be either of the following values:
swaAllVisibleEntities
swaSelectionSet

ExportOffsetFile (Method)Available only with Basic and Advanced Marine Options

Description:

Exports a file containing the offsets of all visible transverse contours in the model. This method can only be called if the offsets view is open.

Syntax:

modelobj.ExportOffsetFile pathname, type

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
type – Can be either of the following values:
swaOFE
swaOF4

ExportPATFile (Method)

Description:

Exports a PAT file of surfaces in the model. Symmetry images are included if the ExportSymmetric property has been set to True.

Syntax:

modelobj.ExportPATFile pathname, EntitiesToExport, IncludeNormals

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
EntitiesToExport - Can be either of the following values:
swaAllVisibleEntities
swaSelectionSet
IncludeNormals - A Boolean expression. When True, surface normals will be exported as well.

ExportPLXFile (Method)

Description:

Exports a PLX file of all the surfaces in the current selection set. Symmetry images are not included even if the ExportSymmetric property has been set to True.

Syntax:

modelobj.ExportPLXFile pathname, Eccentricity, Theta, Thickness, Constraint, Style

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
Eccentricity - A value of type Double that specifies the eccentricity parameter.
Theta - A value of type Double that specifies the orientation of the maximum strain direction with respect to the u-direction.
Thickness - A value of type Double that specifies the plate thickness in the direction of the positive normal.
Constraint - A value of type Integer that specifies a constraint on the expansion calculation and may be one of the following:
-1 for a process that can only expand
0 if no constraint
+1 for a process that can only contract
Style - A value of type Integer that specifies the triangle layout. It can have a value of 0, 1 or 2. Style 2 is strongly recommended.

ExportRULFile (Method)

Description:

Exports a Ruling (RUL) file of surfaces that are rulable. Symmetry images are included if the ExportSymmetric property has been set to True.

Syntax:

modelobj.ExportRULFile pathname, EntitiesToExport

modelobj - An expression that evaluates to an AMSModel object.

pathname - A String expression that specifies a filename and may also include a drive and directory specification.
EntitiesToExport - Can be either of the following values:
swaAllVisibleEntities
swaSelectionSet

ExportSymmetric (Property)

Description:

A Boolean value which when set to True causes symmetry images to be included in any files exported by the Export3DAFile, Export3DDXFFile, ExportIGESFile, ExportPATFile, ExportPLXFile and ExportRULFile methods. Default value is False.

Syntax:

modelobj.ExportSymmetric [= bool]

modelobj - An expression that evaluates to an AMSModel object.

bool - A Boolean expression that evaluates to True or False.

FullName (Read-only Property)

Description:

Returns a String holding the full pathname of the model.

Syntax:

modelobj.FullName

modelobj - An expression that evaluates to an AMSModel object.

Hydrostatics (Method) Available only with Basic and Advanced Marine Options

Description:

Calculates the model's hydrostatics. This method will only work if the Offsets view is open and is the active view.

Syntax:

modelobj.Hydrostatics settings, numsta, sectdata, results
modelobj - An expression that evaluates to an AMSModel object.
settings - A Variant that contains an array of type Single. The array is a one-dimension array with five elements.

Index	Value
1	Specific gravity
2	Z-coordinate of center of gravity
3	Sink
4	Trim
5	Heel

Hydrostatics (cont.)

numsta - An Integer that returns the number of stations used to calculate the hydrostatics.

sectdata - A Variant which returns a two-dimensional array of type Single containing data about the stations. The first dimension of this array refers to the index of the station while the second dimension refers to the data specified in the table below.

Index	Value
1	X-position
2	Immersed area
3	Y-coordinate of centroid
4	Z-coordinate of centroid
5	Wetted girth
6	Width at waterline

results - A Variant which returns a one-dimensional array of type Single containing different hydrostatic data. The following table shows what the contents of the array are according to index.

Index	Value
1	X-position of forward end of waterline
2	X-position of aft end of waterline
3	Waterline length
4	Beam at waterline
5	Draft
6	Immersed volume
7	X-position of center of buoyancy
8	Longitudinal center of buoyancy (as percentage of wl length)
9	Displacement
10	Y-position of center of buoyancy
11	Z-position of center of buoyancy
12	Waterplane area
13	X-position of center of flotation
14	Longitudinal center of flotation (as percentage of wl length)
15	Wetted surface area
16	X-position of center of wetted surface area
17	Z-position of center of wetted surface area
18	Transverse metacentric height (GM_T)
19	Transverse righting moment (RM_T)
20	Longitudinal metacentric height (GM_L)
21	Longitudinal righting moment (RM_L)
22	Waterplane coefficient
23	Prismatic coefficient
24	Block coefficient
25	Midship section coefficient
26	Displacement-length ratio

Example:

```
Dim i As Integer
Dim j As Integer
Dim OpenWarning As Integer
Dim swViews As SWAdvMod.AMSViews
```

```
Dim HydroSettings(1 To 5) As Single
Dim Settings As Variant
Dim NumSta As Integer
```

Hydrostatics (cont.)

```
Dim SectData As Variant
Dim Results As Variant

'Open offsets view
Set swViews = swModel.Views


Dim ViewOpts(1 To 4) As Variant
' By not setting the first element, we are accepting the
' default value for gap.
ViewOpts(2) = True      ' Adds a deck to the offsets. Set to
' false if contours do not need to be
' closed across their tops.
ViewOpts(3) = False     ' Do not add a bottom. Set to true if
' contours need to be closed along
' their bottoms.
ViewOpts(4) = False     ' Do not show points
swViews.Open swaOffsetsView, ViewOpts, OpenWarning


HydroSettings(1) = 64
HydroSettings(2) = 0.75
HydroSettings(3) = 1.175
HydroSettings(4) = 0
HydroSettings(5) = 0
Settings = HydroSettings()


' Now get the hydrostatics.
swModel.Hydrostatics Settings, NumSta, SectData, Results


For i = 1 To NumSta
   For j = 1 To 6
      MsgBox SectData(i, j)
   Next j
Next i
For i = 1 To 26
   MsgBox Results(i)
Next i
```

IsSymmetric (Read-only Property)

Description:

Returns a Boolean which when True indicates the presence of symmetry in the model.

Syntax:

modelobj.IsSymmetric

modelobj - An expression that evaluates to an AMSModel object.

LayerName (Property)

Description:

Sets or returns the layer name for a given layer number.

Syntax:

modelobj.LayerName(number) [= name]

modelobj - An expression that evaluates to an AMSModel object.

number – An expression of type Long which specifies the layer number.
name – A String that specifies the layer name.

LayerState (Property)

Description:

Sets or returns the on/off state of a given layer.

Syntax:

modelobj.LayerName(number) [= state]

modelobj - An expression that evaluates to an AMSModel object.

number – An expression of type Long which specifies the layer number.
state – A Boolean expression. When True, the layer is on. When False, the layer is off.

Name (Read-only Property)

Description:

Returns a String holding the filename of the model, not including the path.

Syntax:

modelobj.Name

modelobj - An expression that evaluates to an AMSModel object.

Parent (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

object = modelobj.Parent

object - An expression that evaluates to an object.

modelobj - An expression that evaluates to an AMSModel object.

Path (Read-only Property)

Description:

Returns a String holding the path of the model, not including the filename.

Syntax:

modelobj.Path

modelobj - An expression that evaluates to an AMSModel object.

Save (Method)

Description:

Saves changes to the model under the path and filename contained in the FullName property.

Syntax:

modelobj.Save

modelobj - An expression that evaluates to an AMSModel object.

SaveAs (Method)

Description:

Saves the contents of the model under the specified filename.

Syntax:

modelobj.SaveAs filename

modelobj - An expression that evaluates to an AMSModel object.

filename - A String expression specifying the filename to save the model under. May also include a path as well.

Saved (Read-only Property)

Description:

Returns a Boolean indicating the saved state of the model. A value of False indicates the model contains one or more unsaved changes.

Syntax:

modelobj.Saved

modelobj - An expression that evaluates to an AMSModel object.

SelectionSet (Read-only Property)

Description:

Returns the AMSSelectionSet object.

Syntax:

modelobj.SelectionSet

modelobj - An expression that evaluates to an AMSModel object.

Symmetry (Property)

Description:

Gets or sets a Variant containing an integer array of model-level symmetry indices.

Index	Value
0	1 if model has any global symmetry, else 0

1	1 if X-mirror symmetry, else 0
2	1 if Y-mirror symmetry, else 0
3	1 if Z-mirror symmetry, else 0
4	1|2|3 for rotational symmetry axis, else 0
5	no. of rotational symmetry images - 1, e.g. 2 for 3-fold symmetry

Symmetry (Cont.)

Syntax:

modelobj.Symmetry [= settings]

settings - A Variant that contains an array of type Integer. The array is a one-dimensional array with six elements.

Units (Property)

Description:

Sets or returns the units being used with a model.

Syntax:

modelobj.Units [= value]

modelobj - An expression that evaluates to an AMSModel object.

value - Can be one of the following:

swaNoUnits
swaFeetAndPounds
swaMetersAndKilograms
swaFeetAndShortTons
swaFeetAndLongTons
swaMetersAndMetricTons
swaInchesAndPounds
swaMillimetersAndGrams
swaCentimetersAndGrams

Views (Read-only Property)

Description:

Returns the AMSViews collection.

Syntax:

modelobj.Views

modelobj - An expression that evaluates to an AMSModel object.

WeightAndCG (Read-only Property)

Description:

Returns a Variant containing an array of data pertaining to the model's weight characteristics. You will only get meaningful data from this property if the model has any entities with weight values attached to them. Each element in the array is a value of type Single. The table below shows what value each element represents.

Index	Value Returned
0	Total weight of model
1	X - Center of Gravity
2	Y - Center of Gravity
3	Z - Center of Gravity

Syntax:

WtCG = modelobj.WeightAndCG

WtCG – A Variant expression containing an array of Singles.

modelobj - An expression that evaluates to an AMSModel object.

AMSModels

Add (Method)

Description:

Creates a new model, adds it to the collection, returns the AMSModel object and makes the model active.

Syntax:

modelsobj.Add

modelsobj - An expression that evaluates to an AMSModels object.

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

modelsobj.Application

modelsobj - An expression that evaluates to an AMSModels object.

Close (Method)

Description:

Closes all currently open models.

Syntax:

modelsobj.Close saveChanges

modelsobj - An expression that evaluates to an AMSModels object.

savechanges – Can be any of the following values:

swaDoNotSaveChanges
swaPromptToSaveChanges
swaSaveChanges

Count (Read-only Property)

Description:

Returns a Long holding the number of currently loaded models.

Syntax:

modelsobj.Count

modelsobj - An expression that evaluates to an AMSModels object.

Model (Method)

Description:

Returns the AMSModel object of a model in the AMSModels collection.

Syntax:

modelsobj.Model(item)

modelsobj - An expression that evaluates to an AMSModels object.

item - A Variant expression specifying a model. It may be a String indicating the name of the model (i.e., its file name with extension) or an integer indicating the index of the model in the collection.

Open (Method)

Description:

Opens an existing model, adds it to the collection, returns the AMSModel object and makes the model active.

Syntax:

modelsobj.Open filename

modelsobj - An expression that evaluates to an AMSModels object.

filename - A String expression specifying the name of the model to open. May also include a path as well.

Parent (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

object = modelsobj.Parent

object - An expression that evaluates to an object.

modelsobj - An expression that evaluates to an AMSModels object.

AMSEntity

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

entity.Application

entity - An expression that evaluates to an AMSEntity object.

BoundingBox (Read-only Property)

Description:

Returns a Variant containing an array of type Double. The array contains the minimum and maximum coordinates of the object. The table below shows what each element in the array represents.

Index	Value
1	Minimum x-position
2	Minimum y-position
3	Minimum z-position
4	Maximum x-position
5	Maximum y-position
6	Maximum z-position

Syntax:

entity.BoundingBox

entity - An expression that evaluates to an AMSEntity object.

Class (Read-only Property)

Description:

Returns a String holding the name of the entity class to which the AMSEntity object belongs. The possible values returned are: “Point”, “Curve”, “Surface”, “Contour”, “Plane”, “Frame”, “Solid” and “No Class”.

Syntax:

entity.Class

entity - An expression that evaluates to an AMSEntity object.

Clearance (Read-only Property)

Description:

Returns a Double holding the Euclidean distance between the entity specified by entity and the entity specified by otherentity. Either entity or otherentity must be a Point entity; the other one can be a point, curve, surface or plane.

Syntax:

entity.Clearance otherentity

entity - An expression that evaluates to an AMSEntity object.

otherentity - An expression that evaluates to an AMSEntity object other than entity.

ClosestPtInSurfTable (Method)

Description:

Finds the point in a surface's table that is closest to the given point. This method can only be used if the AMSEntity object is a surface.

Syntax:

entity.ClosestPtInSurfTable Point, u, v, x, y, z

entity - An expression that evaluates to an AMSEntity object.

ClosestPtInSurfTable (Cont.)

Point - A Variant containing an array of type Double. The array contains the x (index = 1), y (index = 2) and z (index = 3) coordinates of a point.
u - A Double that holds the u parameter of the point on the surface closest to Point.
v - A Double that holds the v parameter of the point on the surface closest to Point.
x - A Double that holds the x coordinate of the point on the surface closest to Point.
y - A Double that holds the y coordinate of the point on the surface closest to Point.
z - A Double that holds the z coordinate of the point on the surface closest to Point.

Color (Property)

Description:

Sets or returns an Integer value specifying the entity’s color.

Syntax:

entity.Color [= number]

entity - An expression that evaluates to an AMSEntity object.

number - A Integer specifying an entity’s new color. Must be between 0 and 255, inclusive. Only 16 colors are used in SurfaceWorks, so the resultant color from number is number/16. This is modulo division so that 0-15 becomes 0, 16-31 becomes 1 etc.

Divisions (Property)

Description:

Sets or returns the entity’s divisions. This property only works with those entities that have divisions, e.g., curves and surfaces. Returns an error if you attempt to get or set the divisions of an entity that does not have a divisions attribute.

Syntax:

entity.Divisions [= vdiv]

entity - An expression that evaluates to an AMSEntity object.

vdiv – A Variant containing an array of integers. The contents of the array are described in the table below.

Entity Class	Index	Description
Curve	1 2	t-divisions t-subdivisions
Surface	1 2 3 4	u-divisions u-subdivisions v-divisions v-subdivisions

Example:

```
' Set a surface's divisions.
Dim vDiv As Variant
Dim NewDiv(1 To 4) As Integer
NewDiv(1) = 10  'u-divisions
NewDiv(2) = 5   'u-subdivisions
NewDiv(3) = 5   'v-divisions
NewDiv(4) = 2   'v-subdivisions
vDiv = NewDiv
entity.Divisions = vDiv
```

EvalCurve (Method)

Description:

Evaluates a curve at given values of t. Returns a Variant containing a two-dimensional array of type Double. The first dimension is the index of an evaluated point on the curve. The second dimension corresponds to the x, y and z coordinates of the point. This method only works if the AMSEntity object's class is a curve.

Syntax:

entity.EvalCurve(tlist)
entity - An expression that evaluates to an AMSEntity object.
tlist - A Variant containing an array of type Double. The array holds a list of t-values at which to evaluate the curve.

Example:

```
Dim varT As Variant
Dim xyz As Variant
Dim t(1 To 2) As Double
t(1) = 0.1
t(2) = 0.25
varT = t()
xyz = entity.EvalCurve(varT)
Dim s As String
s = "Point at t = " & Format(t(1), "###0.000") & ": "
s = s & "x = " & Format(xyz(1, 1), "###0.000") & ", "
s = s & "y = " & Format(xyz(1, 2), "###0.000") & ", "
s = s & "z = " & Format(xyz(1, 3), "###0.000") & vbCrLf
s = s & "Point at t = " & Format(t(2), "###0.000") & ": "
s = s & "x = " & Format(xyz(2, 1), "###0.000") & ", "
s = s & "y = " & Format(xyz(2, 2), "###0.000") & ", "
s = s & "z = " & Format(xyz(2, 3), "###0.000")
MsgBox s
```

EvalPlane (Method)

Description:

Evaluates the point and normal that defines the plane. Returns a Variant containing a two-dimensional array of type Double. The first dimension refers to the point and the normal. The second dimension corresponds to the x, y and z coordinates of the point or normal. So position (1, 1) refers to the x-coordinate of the point, and position (2, 3) refers to the z-coordinate of the normal. This method only works if the AMSEntity object's class is a plane.

Syntax:

entity.**EvalPlane**

entity - An expression that evaluates to an AMSEntity object.

Example:

```
Dim PtNorm As Variant
PtNorm = entity.EvalPlane
Dim s As String
s = "Point: x = " & Format(PtNorm(1, 1), "###0.000")
s = s & ", y = " & Format(PtNorm(1, 2), "###0.000")
s = s & ", z = " & Format(PtNorm(1, 3), "###0.000") & vbCrLf
s = s & "Normal: x = " & Format(PtNorm(2, 1), "###0.000")
s = s & ", y = " & Format(PtNorm(2, 2), "###0.000")
s = s & ", z = " & Format(PtNorm(2, 3), "###0.000")
MsgBox s
```

EvalSurface (Method)

Description:

Evaluates a surface at given values of u and v. Returns a Variant containing a two-dimensional array of type Double. The first dimension is the index of an evaluated point on the surface. The second dimension corresponds to the x, y and z coordinates of the point. This method only works if the AMSEntity object's class is a surface.

Syntax:

entity.EvalSurface(uvlist)

entity - An expression that evaluates to an AMSEntity object.

uvlist - A Variant containing a two-dimensional array of type Double. The array holds a list of u and v parameter values at which to evaluate the surface.

Example:

```
Dim varUV As Variant
Dim uv(1 To 2, 1 To 2) As Double
uv(1, 1) = 0.1
uv(1, 2) = 0.2
uv(2, 1) = 0.25
uv(2, 2) = 0.3
varUV = uv()
Dim xyz As Variant
xyz = entity.EvalSurface(varUV)
Dim s As String
s = "Point at u = " & Format(uv(1, 1), "###0.000")
s = s & ", v = " & Format(uv(1, 2), "###0.000")
```

```
s = s & ": x = " & Format(xyz(1, 1), "###0.000")
s = s & ", y = " & Format(xyz(1, 2), "###0.000")
s = s & ", z = " & Format(xyz(1, 3), "###0.000") & vbCrLf
s = s & "Point at u = " & Format(uv(2, 1), "###0.000")
s = s & ", v = " & Format(uv(2, 2), "###0.000")
s = s & ": x = " & Format(xyz(2, 1), "###0.000")
s = s & ", y = " & Format(xyz(2, 2), "###0.000")
s = s & ", z = " & Format(xyz(2, 3), "###0.000")
MsgBox s
```

FixedFloats (Property)

Description:

Sets or returns a Variant array holding the fixed floating point values (type Double) associated with an entity. The fixed floats are those floating point values that are fixed in number and must always be a part of an entity's definition. The contents of the array vary depending on the entity. Entities not appearing in the following tables do not contain fixed floating-point values.

Point Entity	Index	Description
AbsBead	1	The t parameter value.
AbsMagnet	1 2	The u parameter value. The v parameter value.
AbsRing	1	The t parameter value.
ArcLenBead	1	Arc-length distance along the curve.
ArcLenRing	1	Arc-length distance along the snake.
FramePoint	1 2 3	X-coordinate relative to basis point Y-coordinate relative to basis point Z-coordinate relative to basis point
OffsetPt	1	Offset distance from the magnet upon which the OffsetPt depends.

FixedFloats (Cont.)

Point Entity	Index	Description
PolarPoint	1 2 3	Latitude relative to basis point (degrees) Longitude relative to basis point (degrees) Radial distance from basis point
RelBead	1	The difference between the RelBead's parameter value and the parameter value of its parent.
RelMagnet	1 2	The difference in the u parameter between the RelMagnet and its parent magnet. The difference in the v parameter between the RelMagnet and its parent magnet.
RelRing	1	The difference between the RelRing's parameter value and the parameter value of its parent.
TanPoint	1	Distance along the tangent line from the parent bead.
XYZBead	1	x, y or z location
XYZRing	1	x, y or z location
Curve/Snake Entity	**Index**	**Description**
Helix	1	pitch

	2	total angle
CopySnake	1 2	Scale factor in u,v-space. Rotation angle in u,v-space (degrees)
ProjSnake2	1	Draft angle (degrees)
Surface Entity	**Index**	**Description**
BFitSurf	1	Log (base 10) of tolerance.
RevSurf	1 2	Starting angle of rotation. Ending angle of rotation.
Miscellaneous Entity	**Index**	**Description**
Contours	1 2	Distance of the zero-index contour from the mirror. Contour spacing.
Composite Surface	1	Log (base 10) of tolerance
CvContours	1 2	Parameter value for the zero-index contour. Parameter interval between contour positions

Syntax:

entity.FixedFloats [= vararray]

entity - An expression that evaluates to an AMSEntity object.

vararray - A Variant containing an array of Double values.

FixedIntegers (Property)

Description:

Sets or returns a Variant array holding the fixed integer values (not including divisions and orientation) associated with an entity. The fixed integers are those integer values that are fixed in number and must always be a part of an entity's definition. The contents of the array vary depending on the entity. Entities not appearing in the following table do not contain fixed integer values. Division and orientation integers can be accessed through the Divisions and Orientation properties.

FixedIntegers (cont.)

Point Entity	Index	Description
XYZBead	1	type 1 = bead is located by its x-position 2 = bead is located by its y-position 3 = bead is located by its z-position
XYZRing	1	type 1 = ring is located by its x-position 2 = ring is located by its y-position 3 = ring is located by its z-position
Arc	1 2 3	divisions subdivisions type
BCurve	1 2 3	divisions subdivisions type
BSubCurve	1 2 3	divisions subdivisions B-Spline type of the blending functions.
CCurve	1 2 3	divisions subdivisions C-spline type
FCurve	1 2 3	divisions subdivisions type
Snake Entity	**Index**	**Description**
BSnake	1 2 3	divisions subdivisions type
BSubSnake	1 2 3	divisions subdivisions B-Spline type of the blending functions.
CSnake	1 2 3	divisions subdivisions type
EdgeSnake	1	divisions

	2 3	subdivisions type
UVSnake	1 2 3	divisions subdivisions type

FixedIntegers (cont.)

Surface Entity	Index	Description
ALoftSurf	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	type
BFitSurf	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	B-spline type in u direction.
	7	B-spline type in v direction.
	8	Specifies choice for number of control points to use for fit in the u direction.
	9	Specifies choice for number of control points to use for fit in the v direction.
BlendCtr	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	type
Blister	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	type
BLoftSurf	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	B-spline type
CLoftSurf	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	C-spline type
Fillet	1	u divisions
	2	u subdivisions
	3	v divisions
	4	v subdivisions
	5	orientation
	6	type

FixedIntegers (cont.)

Surface Entity	Index	Description
SuperBlend	1 2 3 4 5 6 7 8 9 10 11 12 13	u divisions u subdivisions v divisions v subdivisions orientation type1 type2 type3 type4 utype ncu vtype ncv
TrimSurf	1 2 3 4 5 6 7	u divisions u subdivisions v divisions v subdivisions orientation type Number of triangles.
Miscellaneous Entity	**Index**	**Description**
Contours	1 2	Index of first contour. Index of last contour.
CvContours	1 2	Index of first contour. Index of last contour.
LtTriMesh	1 2	type (unused) Degree of subdivision
Relabel	1	Type of B-splines used in relabeling graph.
TriMesh	1 2	type (unused) Degree of subdivision

Syntax:

entity.FixedIntegers [= vararray]

entity - An expression that evaluates to an AMSEntity object.

vararray - A Variant containing an array of Integer values.

FixedParents (Property)

Description:

Sets or returns a Variant array holding the fixed parents of the entity. The fixed parents are those entity parents that are fixed in number and must always be a part of an entity's definition. The contents of the array vary depending on the entity.

Point Entity	Index	Description
AbsBead	1	curve
AbsMagnet	1	surface
AbsRing	1	snake
ArcLenBead	1	bead / curve

FixedParents (cont.)

Point Entity	Index	Description
ArcLenRing	1	ring / snake
FramePoint	1 2	point frame
IntRing	1 2 3	ring / snake mirror / surface point
IntRing2	1 2	ring / snake ring / snake
MirrPoint	1 2	basis point mirror
OffsetPt	1	basis magnet
PolarPoint	1 2	point frame
ProjMagnet	1 2 3	basis point magnet / surface mirror / surface
ProjPoint	1 2	basis point mirror
RelBead	1	basis bead
RelMagnet	1	basis magnet
RelRing	1	basis ring
TanPoint	1	bead
XYZBead	1 2 3	curve / bead frame point
XYZRing	1 2 3	snake / ring frame point
Curve Entity	**Index**	**Description**
Arc	1	relabel

	2 3 4	point1 point2 point3
BCurve	1 2,…,n+1	relabel n control points
BSubCurve	1 2,…,n+1	relabel n beads
CCurve	1 2,…,n+1	relabel n control points
FCurve	1 2,…,n+1	relabel n control points
Helix	1 2 3	relabel point line
Line	1 2 3	relabel point1 point2
MirrCurve	1 2 3	relabel curve mirror

FixedParents (cont.)

Curve Entity	Index	Description
PolyCurve2	1 2,…,n+1	relabel n curves
ProjCurve	1 2 3	relabel curve mirror
RelCurve	1 2 3 4 5	relabel curve point1 point2 graph
Snake Entity	**Index**	**Description**
BSnake	1 2,…,n+1	relabel n magnets
BSubSnake	1 2,…,n+1	relabel n beads
CopySnake	1 2 3	relabel snake magnet / surface
CSnake	1 2,…,n+1	relabel n magnets
EdgeSnake	1 2	relabel surface
IntSnake	1 2 3	relabel magnet mirror / surface

	4	point
LineSnake	1 2 3	relabel magnet1 magnet2
ProjSnake	1 2 3 4	relabel basis curve magnet / surface mirror / surface
ProjSnake2	1 2 3 4	relabel basis curve magnet / surface mirror / surface
UVSnake	1 2	relabel magnet
Surface Entity	**Index**	**Description**
ALoftSurf	1 2 3 4	relabel curve1 curve2 curve3
BFitSurf	1	surface
BlendCtr	1 2 3 4,…,n+3	point u-graph v-graph n control curves, n = 3 or 4

FixedParents (cont.)

Surface Entity	Index	Description
Blister	1 2 3 4	relabel snake1 snake2 / magnet / ring curve
BLoftSurf	1 2,…,n+1	relabel n curves
CLoftSurf	1 2,…,n+1	relabel n curves
DevSurf	1 2	bead / curve bead / curve
Fillet	1 2 3 4 5	relabel snake1 snake2 snake3 snake4
MirrSurf	1 2	basis surface mirror
OffsetSurf	1	basis surface
ProcCvSurf	1 2,…,n+1	curve n beads / rings / graphs
ProcPtSurf	1 2,…,n+1	point n magnets

RevSurf	1 2 3	relabel basis curve line
RuledSurf	1 2 3	relabel curve1 curve2
SuperBlend	1 2 3 4 5 6	graph for the u-direction graph for the v-direction curve1 curve2 curve3 curve4
SweepSurf	1 2 3 4 5 6	bead curve1 curve2 graph1 graph2 graph3
TranSurf	1 2	curve1 curve2
TrimSurf	1 2,…,n+1	magnet / surface n snakes
Miscellaneous Entity	**Index**	**Description**
Contours	1 2,…,n+1	mirror / surface n surfaces
CvContours	1 2,…,n+1	curve n surfaces

FixedParents (cont.)

Miscellaneous Entity	Index	Description
Frame3	1 2 3	point1 point2 point3
OffsetPlane	1	plane
Plane2	1 2	point1 point2
Plane3	1 2 3	point1 point2 point3
WireFrame	1	frame / point

Syntax:

entity.FixedParents [= vararray]

entity - An expression that evaluates to an AMSEntity object.

vararray- A Variant holding an array of entities.

FixedStrings (Property)

Description:

Sets or returns a Variant containing an array of strings associated with an entity. The fixed strings are those strings that must always be present in an entity's definition. The contents of the array vary depending on the entity.

Entity	Index	Description
WireFrame	1	Filename of the .3DA or .PAT file.

Syntax:

entity.String [= vStr]

entity - An expression that evaluates to an AMSEntity object.

vStr - A Variant containing an array of strings.

IsContourXYZ (Read-only Property)

Description:

Returns one of the following values indicating the orientation of the contours in a contours class entity. This property applies only if the entity's class is a contour.

swaNonParallelContour Contours are not parallel to any of the axial planes.
swaYZContour Contours are parallel to the YZ-plane.
swaXZContour Contours are parallel to the XZ-plane.
swaXYContour Contours are parallel to the XY-plane.

Syntax:

entity.IsContourXYZ

entity - An expression that evaluates to an AMSEntity object.

IsCurveLine (Read-only Property)

Description:

Returns True if the entity is a straight line within the given tolerance. This property applies only if the entity's class is a curve.

Syntax:

entity.IsCurveLine(Tolerance, StartPt, Tangent)

entity - An expression that evaluates to an AMSEntity object.

Tolerance – A Double expression that specifies the tolerance in determining if the entity is a line.

StartPt – A Variant that contains an array of type Double. The x,y,z coordinates (index 1,2,3, respectively) of the curve's starting point is returned in this argument.

Tangent - A Variant that contains an array of type Double. The x,y,z components (index 1,2,3, respectively) of the unit tangent at the starting point is returned in this argument.

IsEntityInError (Read-only Property)

Description:

Returns True if the entity is currently in error.

Syntax:

entity.IsEntityInError(ErrorNumber)

entity - An expression that evaluates to an AMSEntity object.

ErrorNumber – A Long. If the entity is in error, the error number is returned in this argument.

IsPredefinedEntity (Read-only Property)

Description:

Returns True if the entity is one of SurfaceWorks' internally defined (or predefined) entities.

Syntax:

entity.IsPredefinedEntity

entity - An expression that evaluates to an AMSEntity object.

IsRuledSurface (Read-only Property)

Description:

Returns True if the entity meets the definition of a ruled surface. This property is always True for RuledSurfs and DevSurfs. This property applies only if the entity's class is a surface.

Syntax:

entity.IsRuledSurface

entity - An expression that evaluates to an AMSEntity object.

IsSurfacePlanar (Read-only Property)

Description:

Returns True if the entity is a surface that lies entirely in one plane within the given tolerance. This property applies only if the entity's class is a surface. The corner point and surface normal returned by this property are valid only if the surface is planar. If the surface is NOT planar, these returned values are meaningless.

Syntax:

entity.IsSurfacePlanar(Tolerance, Corner00, Normal)

entity - An expression that evaluates to an AMSEntity object.

Tolerance – A Double expression that specifies the tolerance in determining if the entity lies in a plane.

Corner00 – A Variant that contains an array of type Double. The x,y,z coordinates (index 1,2,3, respectively) of the surface's starting point (u=0, v=0) are returned in this argument.

Normal - A Variant that contains an array of type Double. The x,y,z components (index 1,2,3, respectively) of the unit normal at the starting point are returned in this argument.

Layer (Property)

Description:

Sets or returns an entity's layer. When getting an entity's layer, a Variant is returned containing an array of type Variant. The first element of this array (index = 1), is an integer indicating the layer number. The second element (index = 2), is a string indicating the layer name. If the layer does not have a name, an empty string is returned.

Syntax:

entity.Layer [= layer]

entity - An expression that evaluates to an AMSEntity object.

layer – A Variant that may either be an Integer indicating the layer number or a String indicating the layer name. If an integer is specified, it must be between 0 and 255, inclusive.

Model (Read-only Property)

Description:

Returns the model object to which the entity belongs.

Syntax:

entity.Model

entity - An expression that evaluates to an AMSEntity object.

Name (Property)

Description:

Sets or returns a String specifying the entity's name.

Syntax:

entity.Name [= name]

entity - An expression that evaluates to an AMSEntity object.

name - A String specifying an entity's new name.

Orientation (Property)

Description:

Sets or returns a surface's orientation.

Syntax:

entity.Orientation [= val]

entity - An expression that evaluates to an AMSEntity object.

val - Can be either swaNormal or swaReverse.

Relabel (Property)

Description:

Sets or returns an entity's (typically curves and surfaces) relabel entity.

Syntax:

entity.Relabel [= RelabelEntity]

entity - An expression that evaluates to an AMSEntity object.

RelabelEntity - An AMSEntity object bolonging to the Relabel's class.

Subclass (Read-only Property)

Description:

Returns an Integer indicating the entity subclass to which the AMSEntity object belongs.

Syntax:

entity.Subclass

entity - An expression that evaluates to an AMSEntity object.

The RGSubclass enumeration that is provided in the type library defines constants of the values returned by this property.

Table (Read-only Property)

Description:

Returns a Variant array of type Double. An entity's table is a tabulated representation of that entity. Although every entity has a table, not every entity actually stores any information in it. The returned array always has three dimensions and its contents vary depending on the entity subclass as shown in the table below.

Entity Subclass	Index1 Dimensions	Index2 Dimensions	Index3 Dimensions	Description
Point	1 to 3	0	0	x, y, z coordinates
Bead	1 to 4	0	0	x, y, z coordinates t parameter value on curve
Magnet	1 to 5	0	0	x, y, z coordinates u, v parameter values on surface
Ring	1 to 6	0	0	x, y, z coordinates t parameter value on snake u, v parameter values on surface

Table (Cont.)

Entity Subclass	Index1 Dimensions	Index2 Dimensions	Index3 Dimensions	Description
Curve	1 to 3	0	-1 to nt+1	x, y, z coordinates at each division of the curve, where nt is the number of divisions
Snake	1 to 5	0	-1 to nt+1	x, y, z coordinates and u, v parameter values on the surface at each division of the snake
Surface	1 to 3	-1 to nv+1	-1 to nu+1	x, y, z coordinates at each division of the surface, where nu is the number of divisions in the u direction and nv is the number of divisions in the v direction
Contour	1 to 4	0	1 to nPoints	Pen and x, y, z coordinates along each cut. nPoints is the total number of points. A pen number of zero (“pen up”) signals the beginning of a separate polyline within a Contours entity.

Note that the tables for curves, snakes and surfaces are “bordered”, i.e., the parameter range is from -1/nt to 1+1/nt for curves and snakes rather than 0 to 1. The extra points enable quadratic or cubic interpolation and finite differencing throughout the nominal parameter range of 0 to 1.

Syntax:

entity.Table

entity - An expression that evaluates to an AMSEntity object.

TextRepresentation (Read-only Property)

Description:

Returns a String holding the text representation of an entity. The text representation is the representation of an entity as it appears in the model file.

Syntax:

entity.TextRepresentation

entity - An expression that evaluates to an AMSEntity object.

Triangles (Read-only Property)

Description:

Returns a Long indicating the number of triangles in the TrimSurf. The arguments return other data about the triangles. This property applies to TrimSurfs only.

Syntax:

entity.Triangles(varTriangles, varPoints)

entity - An expression that evaluates to an AMSEntity object.

VarTriangles – A Variant containing an array of type Long. The array has two dimensions. The first dimension corresponds to the index of a triangle and has bounds from one to the number of triangles. The second dimension has bounds from one to three and contains the indexes of points in the varPoints array which are the three vertices of the triangle.

VarPoints – A Variant containing an array of type Double.

Example:

```
Dim vTriangles As Variant
Dim vPoints As Variant
Dim nTri As Long
nTri = entity.Triangles(vTriangles, vPoints)
txt = txt & "Triangles of " & entity.Name & vbCrLf
      For i = 1 To UBound(vTriangles, 1)
    For j = 1 To 3
        txt = txt & Format(vTriangles(i, j), "###") & "    "
    Next j
    txt = txt & vbCrLf
Next i
txt = txt & "Coordinates of triangle vertices" & vbCrLf
For i = 1 To UBound(vPoints, 1)
    For j = 1 To 3
        txt = txt & Format(vPoints(i, j), "###0.000") & "    "
    Next j
    txt = txt & vbCrLf
Next i
```

Type (Read-only Property)

Description:

Returns a String holding the name of the entity type to which the AMSEntity object belongs.

Syntax:

entity.Type

entity - An expression that evaluates to an AMSEntity object.

UserData (Property)

Description:

Sets or returns a String value specifying the entity's user data.

Syntax:

entity.UserData [= str]

entity - An expression that evaluates to an AMSEntity object.

str - A String expression specifying the entity's new user data.

VariableListItems (Property)

Description:

Sets or returns a Variant array holding the contents of a variable length list of an entity. Variable length lists may contain integers, floats, parent entities or any combination of these. The contents of the array vary depending on the entity.

Curve Entity	List Index	Element Index1	Element Index2	Description
BCurve	1	1,…,n	N / A	n control points
BSubCurve	1	1,…,n	N / A	n beads
CCurve	1	1,…,n	N / A	n control points
FCurve	1	1,…,n	N / A	n control points
PolyCurve2	1	1,…,n	N / A	n curves

Snake Entity	List Index	Element Index1	Element Index2	Description
BSnake	1	1,…,n	N / A	n magnets
BSubSnake	1	1,…,n	N / A	n beads
CSnake	1	1,…,n	N / A	n magnets

Surface Entity	List Index	Element Index1	Element Index2	Description
BlendCtr	1	1,…,n	N / A	n control curves, n = 3 or 4
BLoftSurf	1	1,…,n	N / A	n curves
CLoftSurf	1	1,…,n	N / A	n curves
OffsetSurf	1	1	N / A	Offset value at the (u=0, v=0) corner if all four offsets are present or the offset value for all four corners if only one offset value is present.
		(2)		Offset value at the (u=1, v=0) corner.
		(3)		Offset value at the (u=1, v=1) corner.
		(4)		Offset value at the (u=0, v=1) corner.
ProcCvSurf	1	1,…,n	N / A	n beads / rings / graphs
ProcPtSurf	1	1,…,n	N / A	n magnets
SubSurf	1	1	N / A	snake
		2		snake
		(3)		optional snake
		(4)		optional snake

VariableListItems (Cont.)

Surface Entity	List Index	Element Index1	Element Index2	Description
SuperBlend	1	1,…,n	N / A	Control values. The figure shows the placement of the control values on the surface in relation to their corresponding index. ncu = number of control values in the u-direction ncv = number of control values in the v-direction (u=0, v=1) (u=1, v=1) ncv 2*ncv . . . ncu*ncv . . . . 2 ncv+2 . . . ncv*(ncu-1)+2 1 ncv+1 . . . ncv*(ncu-1)+1 (u=0, v=0) (u=1, v=0)
TrimSurf	1	1,…,n	N / A	n snakes

Misc. Entity	List Index	Element Index1	Element Index2	Description
Contours	1	1,…,n	N / A	n surfaces
CvContours	1	1,…,n	N / A	n surfaces
LtTriMesh	1	1,…,nn	1,2,3	x,y,z coordinates of nn nodes (Doubles)
	2	1,…,nt	1,2,3	node indices of nt triangles (Integers)
	3	1,…,nb	N / A	nb indices of break points / lines (Integers) Each node index of a break point is followed by a zero. Each list of node indices of a break line is followed by a zero. A single zero in this list indicates no break points / lines.
ObjectList	1	1,…,n	N / A	n objects
Relabel	1	1,…,n	N / A	n parameter values

VariableListItems (Cont.)

Misc. Entity	List Index	Element Index1	Element Index2	Description
TriMesh	1	1,…,np	N / A	np point entities representing triangle nodes
	2	1,…,nt	1,2,3	node indices of nt triangles (Integers)
	3	1,…,nb	N / A	nb indices of break points / lines (Integers) Each node index of a break point is followed by a zero. Each list of node indices of a break line is followed by a zero. A single zero in this list indicates no break points / lines.

Syntax:

entity.VariableListItems(ListIndex) [= vararray]

entity - An expression that evaluates to an AMSEntity object.
ListIndex - An Integer indicating the list index.
vararray - A Variant holding an array of entities, Doubles, Integers or some combination of them.

Visibilty (Property)

Description:

Sets or returns an Integer value specifying the entity's visibility. This property applies only to those entity types that have a visibility.

Syntax:

entity.Visibility [= number]

entity - An expression that evaluates to an AMSEntity object.

number - An Integer expression specifying an entity's new visibility. The SWACurveVis, SWASurfaceVis and SWAWireframeVis enumerations can be used with this property. The members of these enumerations can be combined by adding them together. Points can only have a visibility of 1 or –1. Entities can be hidden by making the value a negative number.

Weight (Property)

Description:

Sets or returns a value of type Double specifying the entity's unit weight attribute. The definition of unit weight varies by entity class:

for points, it is weight

for curves, snakes, and contours, it is weight / arc length

for surfaces, it is weight / unit area

Note that weight has no meaning for some entities such as relabels, planes etc.

Syntax:

entity.Weight [= wt]
entity - An expression that evaluates to an AMSEntity object.
wt - A Double expression which specifies the entity's weight attribute.

AMSEntities

Add (Method)

Description:

Adds a new instance of an entity to the model and returns the AMSEntity object.

Syntax:

entities.Add(textrep)

entities - An expression that evaluates to an AMSEntities object.

textrep - A String expression specifying the text representation of the new entity. The text representation is the representation of a model entity as it appears in the model file. See "Entities Syntax" on page 10-525 for a description of each entity's text representation.

Example:

```
Entities.Add "ArcLenBead bead1 10 1 2 / curve1 3.66;"
```

This example adds Arc-Length Bead "bead1" with color = 10 and visibility = 1 on layer 2. Bead1's parent is "curve1" and it is at a distance of 3.66 units along this curve.

Application (Read-only Property)

Description:

Returns the Application object.

Syntax:

entities.Application

entities - An expression that evaluates to an AMSEntities object.

Count (Read-only Property)

Description:

Returns a Long specifying the number of AMSEntity objects contained in the collection.

Syntax:

entities.Count

entities - An expression that evaluates to an AMSEntities object.

Entity (Method)

Description:

Returns the AMSEntity object of an entity in the model.

Syntax:

entities.Entity(*item*)

entities - An expression that evaluates to an AMSEntities object.

item - A Variant expression specifying an AMSEntity object. It may be a string indicating the name of the entity or an integer indicating the index of the entity in the collection.

Example:

```
' Get the entity named P22.
Dim entities As SWAdvMod.AMSEntities
Dim entity As SWAdvMod.AMSEntity
Dim strName As String
strName = "P22"
Set entities = swModel.Entities
Set entity = entities.Entity(strName)
```

Since Entity is the default property, the last line could also be written as:

```
Set entity = entities(strName)
```

Model (Read-only Property)

Description:

Returns the AMSModel object to which this collection belongs.

Syntax:

entities.Model

entities - An expression that evaluates to an AMSEntities object.

Parent (Read-only Property)

Description:

Returns the AMSModel object to which this collection belongs.

Syntax:

entities.Parent

entities - An expression that evaluates to an AMSEntities object.

Remove (Method)

Description:

Removes an entity from the model and the collection.

Syntax:

entities.Remove(*item*)

entities - An expression that evaluates to an AMSEntities object.

item - A Variant expression specifying the AMSEntity object to remove. It may be a string indicating the name of the entity, an integer indicating the index of the entity in the collection or the AMSEntity object.

UniqueName (Method)

Description:

Returns a String holding an entity name that is unique among the names of existing entities in the model.

Syntax:

entities.UniqueName(prefix, suffix)

entities - An expression that evaluates to an AMSEntities object.

prefix - A String expression specifying a prefix to add to the name. Pass an empty string if no prefix is desired.

suffix - A String expression specifying a suffix to add to the name. Pass an empty string if no suffix is desired.

AMSSelSet

Append (Method)

Description:

Appends entities to the current selection set.

Syntax:

selset.Append items

selset - An expression that evaluates to an AMSSelSet object.

items - A Variant expression specifying the entities to append to the current selection set. The Variant may be either an array of entities or entity names.

Example:

```
Dim varObjs As Variant
Dim objNames(1 To 3) As String
objNames(1) = entities(4).Name
objNames(2) = entities(5).Name
objNames(3) = entities(6).Name
varObjs = objNames
swSelSet.Append varObjs
```

Application (Read-only Property)

Description:

Returns the Application object.

Syntax:

selset.Application

selset - An expression that evaluates to an AMSSelSet object.

Clear (Method)

Description:

Clears the current selection set of all entities.

Syntax:

selset.Clear

selset - An expression that evaluates to an AMSSelSet object.

IsSupportSet (Read-only Property)

Description:

Returns a Boolean which indicates whether or not the current selection set is a support set.

Syntax:

selset.IsSupportSet

selset - An expression that evaluates to an AMSSelSet object.

Parent (Read-only Property)

Description:

Returns the AMSModel object to which this collection belongs.

Syntax:

selset.Parent

selset - An expression that evaluates to an AMSSelSet object.

Remove (Method)

Description:

Removes one or more entities from the current selection set.

Syntax:

selset.Remove item

selset - An expression that evaluates to an AMSSelSet object.

item - A Variant expression that holds an array of entities to remove from the current selection set. The elements of the array may be of type either Object or strings indicating the entity names.

Example:

```
Dim varObjs As Variant
Dim objNames(1 To 3) As String
objNames(1) = entities(4).Name
objNames(2) = entities(5).Name
objNames(3) = entities(6).Name
varObjs = objNames
swSelSet.Remove varObjs

Dim objs(1 To 2) As Object
Set objs(1) = entities(2)
Set objs(2) = entities(3)
```

```
varObjs = objs
swSelSet.Remove varObjs
```

SelectChildren (Method)

Description:

Appends an entity's children to the current selection set.

Syntax:

selset.SelectChildren generation

selset - An expression that evaluates to an AMSSelSet object.

generation – This can be either swaFirstGeneration or swaAllGenerations.

SelectedEntities (Property)

Description:

Returns a Variant holding an Object array of the currently selected entities or sets the selected entities in the current selection set.

Syntax:

selset.SelectedEntities [= var]

selset - An expression that evaluates to a SelectionSet object.

var - A Variant holding an array of entities or entity names.

Example:

```
Dim varObjs As Variant
Dim Objs(1 To 3) As SWAdvMod.AMSEntity
Set Objs(1) = entities(1)
Set Objs(2) = entities(2)
Set Objs(3) = entities(3)
varObjs = Objs
swSelSet.SelectedEntities = varObjs
```

SelectParents (Method)

Description:

Appends an entity's parents to the current selection set.

Syntax:

selset.SelectChildren generation

selset - An expression that evaluates to an AMSSelSet object.

generation – This can be either swaFirstGeneration or swaAllGenerations.

AMSView

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

view.Application

view - An expression that evaluates to an AMSView object.

Close (Method)

Description:

Closes the view.

Syntax:

view.Close
view - An expression that evaluates to an AMSView object.

Model (Read-only Property)

Description:

Returns the AMSModel object to which the view belongs.

Syntax:

view.Model

view - An expression that evaluates to an AMSView object.

Parent (Read-only Property)

Description:

Returns the AMSModel object to which the view belongs.

Syntax:

view.Parent

view - An expression that evaluates to an AMSView object.

SuppressRefresh (Method)

Description:

Suppresses the visual updating of the view. This method is most useful when it is applied to the model view during the creation of many visible entities at one time through a macro. The updating can be suppressed while creating the entities to avoid the constant flicker and the time it takes to update the view.

Syntax:

view.SuppressRefresh(bSuppress)

bSuppress - If True, suppresses the visual updating of the view.

Title (Read-only Property)

Description:

Returns a String holding the title of the view.

Syntax:

view.Title

view - An expression that evaluates to an AMSView object.

Type (Read-only Property)

Description:

Returns one of the following values indicating the view's type:
swaModelView
swaSurfaceCurvatureView
swaShipLinesView (Basic / Advanced Marine versions only)
swaOffsetsView (Basic / Advanced Marine versions only)
swaMassPropertiesView
swaWeightScheduleView
swaHydrostaticsView (Basic / Advanced Marine versions only)
swaErrorView

Syntax:

view.Type
view - An expression that evaluates to an AMSView object.

AMSViews

ActiveView (Property)

Description:

Returns the AMSView object of the currently active view.

Syntax:

viewsobj.ActiveView

viewsobj - An expression that evaluates to an AMSViews object.

Application (Read-only Property)

Description:

Returns the AMSApplication object.

Syntax:

viewsobj.Application

viewsobj - An expression that evaluates to an AMSViews object.

BGColorForRender (Property)

Description:

Sets or returns a Long that specifies the background color for the shaded view as an RGB color value.

Syntax:

viewsobj.BGColorForRender [= BGColor]

viewsobj - An expression that evaluates to an AMSViews object.

BGColor - A Long expression that specifies the background color as an RGB color value.

Example:

```
Dim iBkColr As Long
iBkColr = RGB(150, 200, 175)
swViews.BGColorForRender = iBkColr
```

Count (Read-only Property)

Description:

Returns a Long holding the number of currently open views for a model.

Syntax:

viewsobj.Count

viewsobj - An expression that evaluates to an AMSViews object.

Model (Read-only Property)

Description:

Returns the AMSModel object to which this collection belongs.

Syntax:

viewsobj.Model

viewsobj - An expression that evaluates to an AMSViews object.

MonochromeForRender (Property)

Description:

Sets or returns a Boolean that specifies how entities appear in the shaded view.

Syntax:

viewsobj.MonochromeForRender [= bMono]

viewsobj - An expression that evaluates to an AMSViews object.

bMono - A Boolean expression. When False, entities are shown in their assigned colors. When True, all entities are shown in the color specified by the MonoColorForRender property.

MonoColorForRender (Property)

Description:

Sets or returns an Integer that specifies the color to use for all entities in the shaded view when the MonochromeForRender property is True.

Syntax:

viewsobj.MonoColorForRender [= color]

viewsobj - An expression that evaluates to an AMSViews object.

color - An Integer expression that specifies one of sixteen colors. Must be between 0 and 15, inclusive. Members of the SWAEntityColor enumeration can be used to set this property.

Open (Method)

Description:

Opens a new view and returns the AMSView object.

Syntax:

viewsobj.Open type, options, warn

viewsobj - An expression that evaluates to an AMSViews object.

type – Must be one of the following values:
- swaModelView
- swaSurfaceCurvatureView
- swaShipLinesView (Basic / Advanced Marine versions only)
- swaOffsetsView (Basic / Advanced Marine versions only)
- swaMassPropertiesView
- swaWeightScheduleView
- swaHydrostaticsView (Basic / Advanced Marine versions only)
- swaErrorView

options - A Variant array. The array specifies different options for the view being opened. The following tables list the options used with each view. If an element is not assigned a value, the default value is taken. Model, mass properties, weight schedule and error views do not have any options.

Surface Curvature			
Index	**Option**	**Type**	**Description**
1	Curvature Type	Integer	The type of curvature to display: 0 = Normal curvature 1 = Gaussian curvature 2 = Mean curvature Default value is normal curvature.

Open (cont.)

Surface Curvature (Cont.)			
Index	**Option**	**Type**	**Description**
2	Angle	Single	Used only with normal curvature. Specifies the angle, in degrees, from the u-direction. Must be in the range –90 to 90, inclusive. Default value is zero degrees.
3	High	Single	The high end of the curvature range to display. Regions with curvatures above this value will appear red. If High and Low are both set to zero, the entire range of curvature will be displayed. Default value is zero.
4	Low	Single	The low end of the curvature range to display. Regions with curvatures below this value will appear blue. If High and Low are both set to zero, the entire range of curvature will be displayed. Default value is zero.

Ship Lines			
Index	**Option**	**Type**	**Description**
1	Body Plan Factor	Double	Used only when Style is set to one. Scale factor for the body plan in relation to the profile and plan views. A factor of one draws all three views at the same scale. A factor of two draws the body plan at twice the scale of the other views. Default value is one.
2	XBody	Double	Used only when Style is set to one, two or three. The x-position at which to divide the forward and aft stations in the body plan. Default value is at the halfway point of the model's extents in the x-direction.
3	Mono-chrome	Boolean	If True, entities are drawn in the color specified by Color. If False, entities are drawn using the colors assigned to them. Default value is False.
4	Color	Integer	Specifies the color to draw all entities with if Monochrome is set to True. Must be between 0 and 15, inclusive. Default value is zero (black). 0 = black 1 = dark blue 2 = dark green 3 = dark cyan 4 = dark red 5 = dark magenta 6 = brown 7 = light gray 8 = dark gray 9 = light blue 10 = light green 11 = light cyan 12 = light red 13 = light magenta 14 = yellow

			15 = white

Open (cont.)

Ship Lines (cont.)			
Index	**Option**	**Type**	**Description**
5	Style	Integer	Specifies what style to use to produce the drawing. The list below shows the valid settings for Style. Default value is 1. 1 = Body plan appears above the profile view and is scaled by the Body Plan Factor in relation to the other views. 2 = Body plan is superimposed on the profile view with the centerline positioned at XBody. 3 = Body plan appears to the right of the profile view. 4 = Only station lifts are drawn. 5 = Only buttock lifts are drawn. 6 = Only waterline lifts are drawn.

Offsets			
Index	**Option**	**Type**	**Description**
1	Gap	Double	Specifies the largest distance between surfaces that will be bridged when constructing the sections for the offsets view.
2	Add Deck	Boolean	When True, closes off any gap between the top edge of all sections and the centerline.
3	Add Bottom	Boolean	When True, closes off any gap between the bottom edge of all sections and the centerline.
4	Show Points	Boolean	When True, all points used to create the sections will be displayed.

Hydrostatics			
Index	**Option**	**Type**	**Description**
1	Specific Weight	Double	Weight per unit volume of water (or whatever liquid the model is being floated in).
2	ZCG	Double	Z-position of the center of gravity.
3	Sink	Double	Vertical position of origin of model coordinates relative to the water surface.
4	Trim	Double	Inclination, in degrees, of model's X-axis relative to the water surface.
5	Heel	Double	Rotation, in degrees, about the model's X-axis.

Open (cont.)

warn - An Integer which returns the warning code of any warnings which occurred while opening the view. Possible values are given in the table below. Currently, the offsets view is the only view that returns a code other than zero. The code returned may be a combination of one or more of the codes listed.

View	Code	Description
All	0	No warning
Offsets	1	Fewer than 10 stations
	2	Max. station spacing > 10% overall length
	4	1st sta. more than 2% overall length from end
	8	Last sta. more than 2% overall length from end
	16	One or more stations have loops with zero enclosed area

Parent (Read-only Property)

Description:

Returns the AMSModel object to which this collection belongs.

Syntax:

viewsobj.Parent

viewsobj - An expression that evaluates to an AMSViews object.

SuppressErrorView (Property)

Description:

Sets or returns a Boolean that specifies whether to suppress the error view or not. This property is used to prevent the error view from opening when errors occur.

Syntax:

viewsobj.SuppressErrrorView [= bSuppress]

viewsobj - An expression that evaluates to an AMSViews object.

bSuppress - A Boolean expression. When True, the error view will not open when errors occur.

View (Method)

Description:

Returns the AMSView object of a view in the collection.

Syntax:

viewsobj.View(item)

viewsobj - An expression that evaluates to an AMSViews object.

item - A Variant expression specifying a view. It may be a string indicating the title of the view or an integer indicating the index of the view in the collection.

Entities Syntax

The names that are used in the user interface to SurfaceWorks (the ones you and I use) are different from the names used internally by SurfaceWorks. Following is a list of all SurfaceWorks entities and their internal names.

Note: Some entities have more than one representation internally, be sure to use the right one.

In the list, the "..." before the "/" indicates optional data (layer, weight, user data).

Points		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Arc-length Bead	ArcLenBead	ArcLenBead name color visibility ... / bead / curve distance ;
Arc-length Ring	ArcLenRing	ArcLenRing name color visibility ... / ring / snake distance ;
Bead	AbsBead	AbsBead name color visibility ... / curve t ;
	RelBead	RelBead name color visibility ... / bead dt ;
Intersection Ring	IntRing	IntRing name color visibility ... / ring / snake mirror / surface point ;
	IntRing2	IntRing2 name color visibility ... / ring / snake1 ring / snake2 ;
Magnet	AbsMagnet	AbsMagnet name color visibility ... / surface u v ;
	RelMagnet	RelMagnet name color visibility ... / magnet du dv ;
Mirrored Point	MirrPoint	MirrPoint name color visibility ... / point mirror ;
Offset Point	OffsetPt	OffsetPt name color visibility ... / magnet offset ;
Point	FramePoint	FramePoint name color visibility ... / type frame point dx dy dz ;
	PolarPoint	PolarPoint name color visibility ... / type frame point lat lon radius ;
Projected Magnet	ProjMagnet	ProjMagnet name color visibility ... / point magnet / surface mirror / surface ;
Projected Point	ProjPoint	ProjPoint name color visibility ... / point mirror ;
Ring	AbsRing	AbsRing name color visibility ... / snake t ;
	RelRing	RelRing name color visibility ... / ring dt ;
Tangent Point	TanPoint	TanPoint name color visibility ... / bead offset ;
XYZ Bead	XYZBead	XYZBead name color visibility ... / type bead / curve frame point location ;

XYZ Ring	XYZRing	XYZRing name color visibility ... / type ring/snake frame point location ;

Entities Syntax (Cont.)

Curves		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Arc	Arc	Arc name color visibility divisions ... / relabel type point1 point2 point3 ;
B-spline Curve	BCurve	BCurve name color visibility divisions ... / relabel type { point1 point2 .... pointN } ;
C-spline Curve	CCurve	CCurve name color visibility divisions ... / relabel type { point1 point2 .... pointN } ;
Foil Curve	FCurve	FCurve name color visibility divisions ... / relabel type { point1 point2 .... pointN } ;
Helix	Helix	Helix name color visibility divisions ... / relabel point line pitch angle ;
Line	Line	Line name color visibility divisions ... / relabel point1 point2;
Mirrored Curve	MirrCurve	MirrCurve name color visibility divisions ... / relabel curve mirror ;
PolyCurve	PolyCurve2	PolyCurve2 name color visibility divisions ... / relabel { curve1 .... curveN } ;
Projected Curve	ProjCurve	ProjCurve name color visibility divisions ... / relabel curve mirror ;
Relative Curve	RelCurve	RelCurve name color visibility divisions ... / relabel curve point1 point2 graph ;
SubCurve	BSubCurve	BSubCurve name color visibility divisions ... / relabel type { bead1 bead2 .... beadN } ;

Snakes		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
B-spline Snake	BSnake	BSnake name color visibility divisions ... / relabel type { magnet1 magnet2 .... magnetN } ;
C-spline Snake	CSnake	CSnake name color visibility divisions ... / relabel type { magnet1 magnet2 .... magnetN } ;
Copy Snake	CopySnake	CopySnake name color visibility divisions ... / relabel snake magnet / surface scale angle ;
Edge Snake	EdgeSnake	EdgeSnake name color visibility divisions ... / relabel type surface ;
Intersection Snake	IntSnake	IntSnake name color visibility divisions ... / relabel magnet mirror / surface point ;

Line Snake	LineSnake	LineSnake name color visibility divisions ... / relabel magnet1 magnet2 ;

Entities Syntax (Cont.)

Snakes (Cont.)		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Projected Snake	ProjSnake	ProjSnake name color visibility divisions ... / relabel curve magnet / surface mirror / surface ;
	ProjSnake2	ProjSnake2 name color visibility divisions ... / relabel curve magnet / surface mirror / surface angle ;
SubSnake	BSubSnake	BSubSnake name color visibility divisions ... / relabel type { bead / ring1 bead / ring2 .... bead / ringN } ;
UVSnake	UVSnake	UVSnake name color visibility divisions ... / relabel type magnet ;

Surfaces		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Arc Lofted Surface	ALoftSurf	ALoftSurf name color visibility divisions orientation ... / relabel type curve1 curve2 curve3 ;
B-spline Lofted Surface	BLoftSurf	BLoftSurf name color visibility divisions orientation ... / relabel type { curve1 curve2 .... curveN } ;
Blend Surface	Fillet	Fillet name color visibility divisions orientation ... / relabel type snake1 snake2 snake3 snake4 ;
Blister	Blister	Blister name color visibility divisions orientation ... / relabel type snake1 snake2 curve;
C-spline Lofted Surface	CLoftSurf	CLoftSurf name color visibility divisions orientation ... / relabel type { curve1 curve2 .... curveN } ;
Centerpoint Boundary Surface	BlendCtr	BlendCtr name color visibility divisions orientation ... / type point graph1 graph2 { curve1 curve2 curve3 (curve4) } ;
Developable Surface	DevSurf	DevSurf name color visibility divisions orientation ... / curve1 curve2 ;
Fitted Surface	BFitSurf	BFitSurf name color visibility divisions orientation ... / utype vtype ncu ncv logtolerance surface ;
Mirrored Surface	MirrSurf	MirrSurf name color visibility divisions orientation ... / surface mirror ;
Offset Surface	OffsetSurf	OffsetSurf name color visibility divisions orientation ... / surface { offset1 (offset2) (offset3) (offset4) } ;

Procedural Surface	ProcPtSurf	ProcPtSurf name color visibility divisions orientation ... / point { magnet1 magnet2 .... magnetN } ;
	ProcCvSurf	ProcCvSurf name color visibility divisons orientation ... / curve { bead / ring / graph1 bead / ring / graph2 .... bead / ring / graphN } ;

Entities Syntax (Cont.)

Surfaces (Cont.)		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Revolution Surface	RevSurf	RevSurf name color visibility divisions orientation ... / relabel curve line angle1 angle2;
Ruled Surface	RuledSurf	RuledSurf name color visibility divisions orientation ... / relabel curve1 curve2 ;
SubSurface	SubSurf	SubSurf name color visibility divisions orientation ... / { snake1 snake2 snake3 (snake4) } ;
Sweep Surface	SweepSurf	SweepSurf name color visibility divisions orientation ... / bead curve1 curve2 graph1 graph2 graph3 ;
Tangent Boundary Surface	SuperBlend	SuperBlend name color visibility divisions orientation ... / type1 type2 type3 type4 ugraph vgraph curve1 curve2 curve3 curve4 utype ncu vtype ncv { value1 value2 .... valueN } ;
Translation Surface	TranSurf	TranSurf name color visibility divisions orientation ... / curve1 curve2 ;
Trimmed Surface	TrimSurf	TrimSurf name color visibility divisons orientation ... / type no. triangles magnet { snake1 sanke2 ... snakeN } ;

Miscellaneous		
SurfaceWorks Entity	**RG Entity (or Entities)**	**Syntax**
Contours	Contours CvContours	Contours name color visibility ... / mirror first_index last_index q0 qint { surfaces } ; CvContours name color visibility ... / curve first_index last_index t0 tint { surfaces } ;
Entity List	ObjectList	ObjectList name ... / { entities } ;
Frame	Frame3	Frame3 name color visibility ... / point1 point2 point3 ;
Plane	OffsetPlane Plane2 Plane3	OffsetPlane name color visibility ... / plane offset ; Plane2 name color visibility ... / point1 point2 ; Plane3 name color visibility ... / point1 point2 point3 ;
Relabel	Relabel	Relabel name ... / type { 0. value2 value3 .... 1. } ;
Wireframe	WireFrame	WireFrame name color visibility ... / filename frame ;

Writing Your First Macro

(from a June 1997 AeroHydro newsletter article written by Michael Shook)

You've used SurfaceWorks, you love it, but you've found some operations you repeat again and again. You wish there were some way to automate the operation (this is a computer after all) and save yourself a lot of time. You're in luck! SurfaceWorks 6.0 exposes much of its functionality to controller programs written using programming languages which support Microsoft's OLE Automation standard.

To demonstrate the capability I've written a short Visual Basic (VB) program which adds a useful capability to SurfaceWorks: given a curve in the Selection Set, it displays a dialog box with the name and length of the curve. Of course you can always find the length of a line with **Tools>Mass Properties**, but there you have to sift through other information, quite extensive in a large model. If you don't have Visual Basic, don't despair, the code I develop here works unchanged as an Excel macro. The techniques, if not the code, will work with any language which supports OLE Automation.

If you've used SurfaceWorks some (for instance, you've gone through the tutorials) and you've ever written a program, you should be able to follow my presentation below. If you've never written a program, but you think you'd like to try this, run through the tutorials that come with Visual Basic first, and then come back and give this a try. So here we go…

The Windows environment usually provides many different ways to do the same thing, and that is definitely true here. I'm just going to show you one way. Here's how I proceeded.

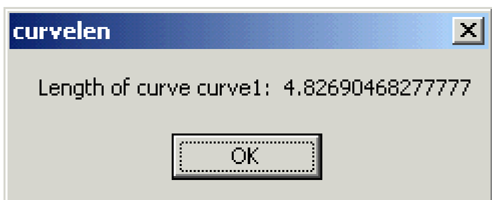
In Visual Basic I create a new Visual Basic project (.VBP) and name it "curvelen". Then I create a new module (a file that only contains VB code, no form) and insert a "sub" procedure into the module and call it "Main". Be sure to add the SurfaceWorks Type Library to the project references. It should be listed as **AeroHydro** SurfaceWorks **Advanced Modeling 1.0 Type Library**.

Now that Visual Basic is set up, I can proceed with the code. Lines 2 and 3 declare and initialize the application object which I use as the starting point to access all other SurfaceWorks Automation objects. SurfaceWorks must already be running for GetObject to work. Lines 4 and 5 get an array of all the selected SurfaceWorks entities and then get the 'table' for the first entity. Every SurfaceWorks geometric entity has a table of points associated with it. For curve entities, the table consists of a one-dimensional set of points located on the curve. The number of points in the table is determined by the curve's divisions and subdivisions - specifically there are (divisions X subdivisions)+1 points.

```
1  Public Sub Main()
2  Dim App As SWAdvMod.AMSApplication
3  Set App = GetObject(, "SWAdvMod.Application")
4  SelEnts = App.ActiveModel.SelectionSet.SelectedEntities
5  Table = SelEnts(1).Table
6  For i = LBound(Table, 3) To UBound(Table, 3) - 1
```

```
7      dx = Table(1, 0, i + 1) - Table(1, 0, i)
8      dy = Table(2, 0, i + 1) - Table(2, 0, i)
9      dz = Table(3, 0, i + 1) - Table(3, 0, i)
10     Length = Length + Sqr(dx * dx + dy * dy + dz * dz)
11 Next i
12 Msg$ = “Length of curve “ + SelEnts(1).Name
13 Msg$ = Msg$ + “: “ + Str$(Length)
14 Response = MsgBox(Msg$)
15 End Sub
```

The table contains everything I need to compute the length of the curve. My program is finished getting information from SurfaceWorks. The loop in lines 6 through 11 calculates the curve length by using the Pythagorean theorem to calculate the length of each line segment and summing those individual lengths. In lines 12 through 14, I build and display the answer - including the curve’s name obtained via the Name property in line 12.

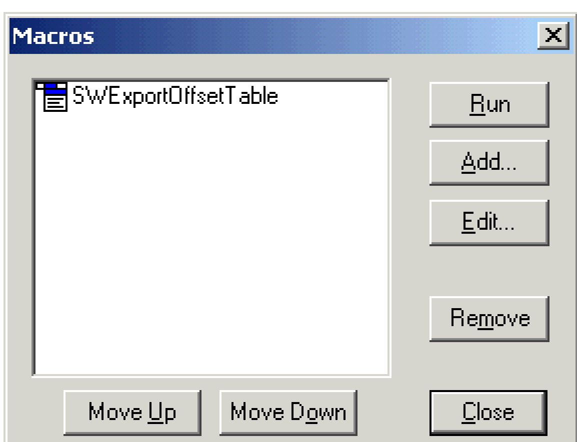


There it is! Curve name and length without searching through mass properties. Once I’ve entered the code into VB, I test it by selecting any curve or snake in a SurfaceWorks model and then running my VB program via **Run>Start**. When I have it running just the way I want it, I compile it with **File/Make EXE File**, then switch to SurfaceWorks and add the EXE as a macro to my SurfaceWorks tools menu using **Tools/Macros**.

Macros Dialog

Tools>Macros opens the macros dialog box.

SurfaceWorks



To run a macro, click on the macro in the window, and click **Run**.

To add a macro to the list, click **Add** and follow the instructions for Macro Properties.

To edit macro properties, click **Edit**.

To remove a macro from the list click **Remove**.

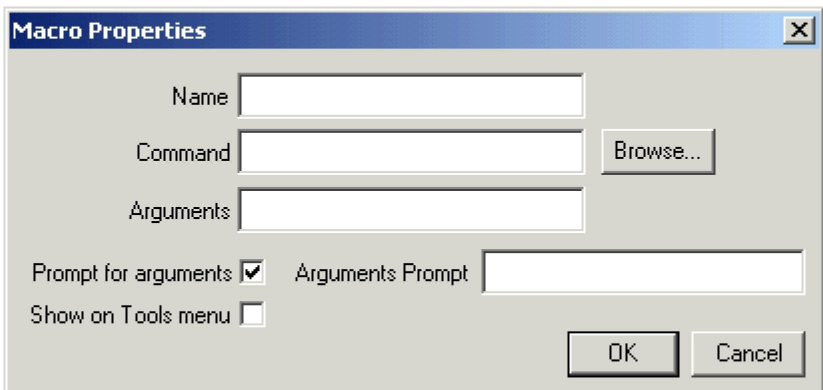
To change the order of the list use the **Move Up** or **Move Down** buttons.

Running a Macro from SurfaceWorks

To run a macro from within SurfaceWorks, the macro must be written to be run from within SurfaceWorks see "Getting the MultiSurf Application Object" on page 468. Go to **Tools>Macros** and click on the <**Add**> button, and the **Macro Properties** box appears.

Macro Properties

1 Fill in the **Name** as you want it to appear in SurfaceWorks, or if you do not enter a name, it will default to "**MacroName**" when you select the macro "**MacroName.exe**" with the browser.



2 Fill in the **Command** box with the name of your macro executable (.exe) with full path, or browse for it using the browse button. If you use the **Browse** button, double click on the executable or highlight the desired file and click **Open**. SurfaceWorks places "**MacroName**" in the **Name** field (if you left the name field blank) and something like **C:\Program Files\AeroHydro\ Macros\ MacroName.exe** in the **Command** field.

3 If your macro requires argument(s) that will be used every time you run it, type them in the **Arguments** box. These arguments should appear here exactly as they would on the command line.

4 If there are arguments that could change, use the **Prompt for arguments** check box, and in the **Arguments Prompt** box enter a mnemonic to remind you of the arguments you will need to run this macro. In this case, only a mnemonic is required, as SurfaceWorks will open a command line for you to type in the arguments at run time, and what you type in the **Arguments Prompt** box appears as a reminder.

5 If you want the macro to appear on the **Tools** menu, click on the **Show on Tools menu** checkbox.

6 Click <**OK**> to close the Macro Properties dialog box.

7 Click <**Close**> to exit the Macros dialog box.

8 To run the macro, choose **Tools>Macros**, pick the macro you want from the list and click **Run**, or if you chose to place the macro on the **Tools** menu, choose it directly from the **Tools** menu.

Export Offset Table Macro instructions

Introduction

The macro SWExportOffsetTable.exe automates the process of creating a standard table of offsets for a SurfaceWorks model. This replaces the old OFTBL utility, which was previously used by SurfaceWorks.

To use the Export Offset Table macro

Set Up

1 Open SurfaceWorks.

2 Go to **Tools>Macros**.

3 Click on the <**Add**> button.

Name:

Fill in the name, or browse for the command. (If you do not enter a name, it will default to **SWExportOffsetTable** when you select the command with the browser).

Command:

Browse for **SWExportOffsetTable.exe**. This executable will appear in the Macros\Export Offset Table folder of your install directory, or wherever you decided to put it if you received it on a separate disk or downloaded it from the web.

Double clicking **SWExportOffsetTable.exe** will put **SWExportOffsetTable** in the <**Name**> field (if you left the name field blank) and something like **C:\Program Files\ AeroHydro** SurfaceWorks\ **Macros\Export Offset Table\ SWExportOffsetTable.exe** in the <**Command**> field.

Arguments:

There are no arguments required for this command.

4 If you want the macro to appear on the **Tools** menu, click on the **Show on Tools menu** checkbox.

5 Click <**OK**> to close the Macro Properties dialog box.

6 Click <**Close**> to exit the Macros dialog box.

Use

1 Select the Contours, Curve and / or Snake entities that are to appear in the offset table.

Note: You must select at least one Contours entity perpendicular to the X-axis plus either a Contours entity perpendicular to the Y- or Z-axis or a curve or a snake.

2 Choose **Tools>SWExportOffsetTable** (or choose **Tools>Macros**, highlight SWExportOffsetTable, then click <**Run**>.

3 In the dialog, set offset table options.

4 Click **OK**.

5 If you chose either the Save text file or Save DXF file Output Option a dialog box appears for you to enter the file name.

6 A progress meter will display while the macro is working.

Offset table options

Title – Text identification of the model. The title will appear in the header portion of the table. The first line of the model's comment is offered as the default.

X-Position of Station 0 and Station Spacing

The Export Offset Table macro creates the table of offset values from the Contours entity(s) that are perpendicular to the X-axis. These offset values are not affected by the values of either X-Position of Station 0 or Station Spacing.

X-Position of Station 0 or Station Spacing are only used to establish the numbering of the stations in the offset table. Generally one creates Contours entities that match the desired station location and spacing, if this is done, the default values will probably work fine. If more than one Contours entity perpendicular to the X-axis is used, you may have to adjust either X-Position of Station 0 or Station Spacing or both in order to get the station numbering to be what you desire.

X-Position of Station 0 – The X-position where Station 0 is located, in the same units as the model. The default value is taken from the value given in the first selected contours entity that is perpendicular to the X-axis.

Station Spacing – The distance between stations, in the same units as the model. The default value is taken from the value given in the first selected Contours entity that is perpendicular to the X-axis.

Y-position of centerline – The Y-position from which the half breadths will be referenced. Default is zero. A different value may be useful for **multihulls** where one hull is created offset from the Y=0 plane in SurfaceWorks and you want to reference the half breadths for that hull from its own center.

Z-position of baseline – The Z-position from which the heights will be referenced. Default is zero.

Format – The format of the numbers that will be used in the table. The default is Decimal, which means the table will show feet and tenths of feet or centimeters and tenths of centimeters, etc. If your model units are feet, you have the additional option of having table format be in feet-inches-8ths. (This is grayed out if you do not have feet as model units). If your model units are unspecified, the macro looks at your Windows Regional Settings and assumes feet for U.S. and meters for Metric.

Offsets to – The part of the boat the offsets represent. This will appear in the table's header. The default is Unspecified.

Curves & Snakes – Lists the curves and snakes selected to appear in the table. The first column shows the names of the curves and snakes, as they will appear in the table. The default names for the table are taken from the names given in the model. You can edit these names (slowly double-click the name, type the new name, press <**Enter**>). The second column shows the names of the curves and snakes as they are given in the model (these names are not editable in this dialog).

Output Options:

Excel 97 worksheet – You must have Excel 97 installed for this option to work. The option opens Excel and creates the offset table in a worksheet. Once the table is generated, you can modify its appearance to suit your own tastes.

Save text file – Saves the offset table in an ASCII text file format. You have the opportunity to specify file name and location in the Save As dialog.

Save text file formatted for tiled paper – This creates the same file output as was created by the SurfaceWorks OFTBL utility. It is an ASCII text file that is formatted to be printed out using tiled paper. You have the opportunity to specify file name and location in the Save As dialog.

Save comma-delimited text file – Saves an ASCII text file where each item in the table is separated by a comma. This file may be useful for importing into spreadsheet programs. You have the opportunity to specify file name and location in the Save As dialog.

Save DXF file – Saves a DXF file of the table, which can then be imported, into a CAD program. The text is output at a height of one unit, so it is likely that the table will have to be resized after it is imported to the desired scale. You have the opportunity to specify file name and location in the Save As Dialog. Note: These DXF files read into AutoCAD 14; some other DXF viewers may have problems.

Print offset table – Prints the offset table directly to the printer. Set line widths for the table in Printed Output Options (below). After you have clicked OK to this dialog, the print dialog will appear, enabling you to select such options as the number of copies, paper size, and paper orientation.

Printed Output Options

(grayed out unless you have turned on the Print Offset Table radio button in the Offset Options)

Thick/Thin line width – Allows you to select line widths for the lines appearing in the table in printed output. Defaults are 6 and 1. Thick width can be between 1 and 20. Thin width can be between 1 and 10. The relation between the numbers and the physical printed widths of these lines may vary from printer to printer. You will have to discover the widths that work best for your own printer.

Commands

Introduction

Tools>Command Window opens a dialog box through which SurfaceWorks can be operated to a limited extent by typed-in commands, or by a command sequence read from a file. Most commands provide functionality that is new and therefore does not yet exist on a menu, or functionality that is powerful and valuable to small number of users, but not enough to warrant putting on a menu.

Command sequences read from a file permit SurfaceWorks to be accessed and operated by other Windows and DOS programs. Communication is through the file system rather than DDE or OLE protocols.

Disclaimer

The commands that appear in this chapter are supported, and will continue to be supported, by AeroHydro. Support may include moving the functionality to one of the SurfaceWorks main menus.

Occasionally, commands are created by AeroHydro that are diagnostic or experimental functions. Because these commands are of limited interest to most users, they will not be documented here, and they may change or disappear in future program versions.

Interface

The Help command

The command window interface is internally documented by the Help command:

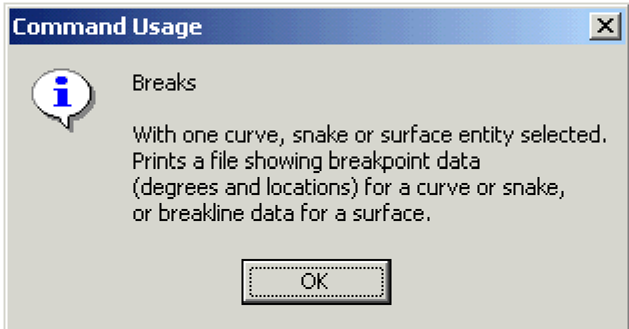
help

by itself displays a list of all the available commands.

help command

displays a message box specifying the command syntax, any required or optional inputs on the command line, selection set needed (if any), and a brief description of what the command does. For example, entering the command:

"help breaks" gives this information for the Breaks command:



Note: Some arguments are case sensitive; e.g. if the argument is a kind of entity (e.g. BSnake).

Command Dropdown

The command dialog has a dropdown that lists the last 20 commands that have been entered. You can click on any one of them to put it into the command window, either for re-execution or as a base for typing in changes.

'Silent' box

Commands often have the ability to send an error message or other information to the screen. If you do not want these messages to appear during the running of the command, click on the 'Silent' check box, and screen messages will be turned off for the duration of the command. Any messages that would have appeared on the screen are written to the file **%cmdmsg.txt** instead, which will be in the directory where WindSurf.exe is located. This file is overwritten if another command is run with the 'Silent' box checked, and a message is sent.

Spaces in paths and filenames

To handle path names and file names with spaces, use double quotes around the path or file name. For example:

setpath “c:\temp\test space”

file.export3d.pat “demo 3A”

Command-line script operation

Since SurfaceWorks 6.5 has a new option to run a script of commands from the MS-DOS command line. This allows SurfaceWorks to function as a geometry server in a system of programs, for example for automated design, analysis, and optimization.

Command scripts

The script syntax used is identical to the syntax for commands in Tools/ Command Window. To see a list of the available commands, open the Command Window and give the command:

Help

To see the syntax for an individual command, use "Help command-keyword"; for example:

Help ExpandLists

A command script is stored as text file of commands. Notepad is a convenient application for preparing and editing text files.

Example. This file, named cmd.txt, and located in c:\TEMP, is a valid command script:

```
SetPath c:\
File.Open demo.ms2
SetFloat P24 3 -1.300
File.Export3D.3DA
File.Close
```

This sets the working directory to c:\ (the root directory of drive c:); opens a copy of demo.ms2 (that must exist in this folder); modifies the geometry by moving a control point; exports a 3DA file (using the default name, Demo.3DA); and closes the model file.

Script execution

First, if SurfaceWorks is already running, close the program. (It can't start a second instance when one is already running.)

Open an MS-DOS command window, and navigate to where your SurfaceWorks executable lives. (In a normal installation this is Program Files\AeroHydro\SurfaceWorks The executable is named Windsurf.exe. Give the DOS command:

windsurf /cmdfile="c:\temp\cmd.txt"

SurfaceWorks will start up (briefly displaying an unconfigured startup screen), execute the designated script, and will then shut itself down.

Following execution of the script, you can confirm that c:\ contains a new file Demo.3DA, with the date and time that the script was run. Since there was a SetPath command, there will also be a %cmdmsg.txt file that echoes the subsequent commands, and could contain useful diagnostic information.

Commands

All supported commands are listed in this section. When a command has arguments, they are separated by a space. If you want to include the second argument, but the default for the first is OK, you must still specify the first argument. For instance, if you want to use CreateCopies with the default color, but use parent 3, you would type CreateCopies 0 3.

Animate

Animate entity_1 period_1 [entity_2 period_2 ...]

Animates a properly constructed model, for 60 seconds.

Each "entity" parameter is the name of a driver entity. Absolute Bead, Absolute Ring and Variable entities currently qualify as drivers.

Each "period" parameter is a time interval in seconds.

During the animation, the t parameter of each Bead or Ring driver is continuously incremented in proportion to elapsed time / period; the model is updated and then redisplayed. When any t parameter passes 1, it snaps back to 0. A Variable driver with limits moves from the lower limit to the upper limit in one period, and snaps back to the lower limit. A Variable with no limits goes from 0 to 1 in one period.

Only shaded and wireframe views can be animated.

A number of animated models are included in the Examples folder: Drip.ms2, Flutter.ms2, Glider.ms2, Pistons.ms2, Pitchpole.ms2, Slalom.ms2, Solar.ms2, Swim.ms2, Swim2.ms2, Tack.ms2. In each case, the last line of the model file is a suitably composed Animate command with appropriate periods.

Edit / Model file (Cancel the optional save)

Copy the last line into your clipboard

Cancel the Edit / Model file dialog

Tools / Command Window

Paste in the Animate command, and OK

Breaks

Breaks []

With one curve, snake or surface in the Selection Set, Breaks displays information about the breakpoints of a curve or snake, or the breaklines of a surface. The message box shows up to ten breakpoints (the degree and t location). A text file “%brkpts.txt” is saved with the same information, but without the ten breakpoint limit.

Breakpoints

SurfaceWorks stores breakpoints for all types of curves and snakes. Breakpoints are taken into account during evaluation, so curves neither “cut the corners” at breakpoints, nor depend on carefully selected divisions, but instead are evaluated with accuracy similar to the rest of the curve.

A **degree-1 breakpoint** is a place on a curve where the slope and / or velocity is discontinuous. For example, degree-1 breakpoints occur at each junction of a PolyCurve, unless the component curves happen to have the same direction and velocity. Degree-1 breakpoints are typical at the control points of a type-1 B-spline Curve or C-spline Curve (i.e., a polyline).

Example of degree-1 breakpoints: In BSPL1-3.MS2 (found in you tutorials folder), ‘type1’ has degree-1 breakpoints at t = 0.25, 0.50, 0.75.

A **degree-2 breakpoint** is a place on a curve where slope and velocity are continuous, but curvature is discontinuous. For example, a place where a line joins a circular arc with tangency is a breakpoint of degree 2 if the velocity is the same on the line and arc. Degree-2 breakpoints are typical at the knots of a type-2 B-spline Curve or C-spline Curve.

Example of degree-2 breakpoints: In BSPL1-3.MS2 (found in your tutorials folder), ‘type2’ has degree-2 breakpoints at t = 1/3 and 2/3.

Breaklines

Breaklines are to surfaces as breakpoints are to curves and snakes. A breakline is a u = constant or v = constant line on a surface where there is a discontinuity of slope, velocity or curvature. SurfaceWorks recognizes and stores breaklines on all surface types. Rather than cutting corners at a breakline, or relying on carefully selected divisions, we take breaklines fully into account.

Breaklines have degrees, similar to breakpoints. A degree-1 breakline is a discontinuity in slope (a crease or knuckle line) or velocity. A degree-2 breakline is a discontinuity of curvature only.

Breaklines in surfaces are mostly inherited from their parent curves; for example, a C-spline Lofted Surface will have a u-breakline wherever any of its master curves has a breakpoint.

Example of degree-1 breaklines: surface ‘hull2’ in Procpsf1.ms2 has two degree-1 u-breaklines. (To see 'hull2', turn on layer 2). The breaklines are located at the junction of 'topside' with 'flat' and 'flat' with 'bottom'.

Example of degree-2 breaklines: surface 'hull1', also in Procpsf1.ms2, has a degree-2 u-breakline. (To see 'hull1', turn on layer 1). 'hull1' is a C-spline Lofted Surface created from three degree-2 C-spline Curves. Each of the C-spline Curves has a degree-2 breakpoint at 0.5, and this creates a degree-2 breakline in 'hull1'.

It is possible to have a knuckle line or crease that does not lie along a constant-u or constant-v line. Such a line is not recognized as a breakline in SurfaceWorks. For example, change ‘hull1’ to Degree-1. This implies a knuckle line along MC2 at roughly v = 0.5, but the v is not constant, varying from 0.512 to 0.444 as u varies from 0 to 1. (Turn on the v-constant lines under **Display>Visibility** for the surface to make this clear.) This knuckle is not recognized as a breakline. In the display of the surface, you can see that the knuckle appears rounded off. In this situation it would be better to make the surface as a type-1 B-spline Lofted Surface, or two Ruled Surfaces, to accurately represent the knuckle.

Countpanels

CountPanels

Counts the panels on visible surfaces.

Create

Given a complete syntax string, e.g.

```
Create BCurve MC2  11  1 10x1 / * 2 { P21 P22 P23 P24 } ;
```

this command creates a new entity from the given string, just as if the string had been read from a model file. The syntax has to be correct, and any supporting entities must already be present, otherwise the create operation is aborted.

CreateCopies

CreateCopies [colorvis [support_num]]

With one “pattern entity” plus N new parents in the Selection Set, CreateCopies creates N new entities, each being a copy of the pattern entity, but with one of the new parents replacing a specified parent in the pattern entity.

colorvis takes two possible values, 0 or 1.

If *colorvis* = 0, (the default value), each copy takes its color, visibility and orientation from the pattern entity. If *colorvis* is 1, it takes these values from the new support entity.

support_num is a value from 0 to the pattern entity’s total number of parents.

If *support_num* is positive, it specifies which one of the pattern entity’s parents will be replaced: 1 = first, 2 = second, etc. If *support_num* is 0 or left blank, (the default value), the new parent replaces the first parent of the pattern entity, but skipping over a leading Relabel, if one is present.

For a pattern entity with only one parent, only that one parent can be replaced.

If a pattern entity has a parent that is a list, i.e. Arc, then only the first parent in the list can be replaced.

Example

SAILBOAT.MS2 (in the Tutorials folder) has 11 surfaces. Suppose we want to create 11 Mirrored Surfaces to make the other half of the boat. First make a Mirrored Surface (say ‘hull1’) from ‘hull’, reflected in the predefined plane ‘*Y=0’. ‘hull1’ will be the pattern entity; the other 10 surfaces (excluding ‘hull’) will be the new parents. Select ‘hull1’ and the other 10 surfaces (with ‘hull1’ first), go to Tools>Command Window, and give the command:

CreateCopies 1

The other 12 Mirrored Surfaces are created. Because we used *colorvis* = 1, the Mirrored Surface colors and visibilities are taken from the parent, not the pattern entity. CreateCopies works with any entity type that has parents. Here’s a list of entity types for which we’ve found CreateCopies especially useful:

Points	Curves/Snakes	Surfaces	Other
Mirrored Point	Mirrored Curve	Mirrored Surface	Offset Plane
Offset Point	Projected Curve	Offset Surface	
Projected Point	Copy Snake		
Tangent Point	Edge Snake		
Bead	Intersection Snake		
Arc-length Bead	Projected Snake		
XYZ Bead	UVSnake		
Magnet			
Projected Magnet			

Ring			
Arc-length Ring			
XYZ Ring			

CreateCopies tutorial

Here's a quick tutorial, which demonstrates a use for CreateCopies. In this example, 2D text, from an outside source, is wrapped on a 3D face. The job is made very simple and quick with the CreateCopies command.

1 Open model AHIWrap.ms2 from the SurfaceWorks Examples folder.

2 Select the blue Ruled Surface '2D_Host'.

3 Zoom to selection or F10

4 Select the visible dark magenta curve 'xedge3D~26'. This is the right hand vertical portion of the letter 'I'.

5 Ctrl-Select the '2D_Host' surface, then select it a second time by pressing Ctrl+Shift while clicking.

6 With these 3 items in the selection set, Insert>Snake>Projected Snake. <OK>.

7 With the new snake, 'n6' in the Selection Set, go to the Home view and Ctrl+click the red surface '3D_Host'.

8 Insert>Snake>Copy Snake. <OK>. Snake 'n7' is created.

9 Select the blue Ruled Surface '2D_Host'. Zoom to selection or F10

10 Select the Projected snake 'n6'.

11 Turn off the Curve filter. Invert the filters so only curves can be selected.

12 With Ctrl pressed, drag a box around all the curves which make up the text string. Take the original curve 'xedge3D~26' out of this Selection Set.

13 The first object selected, 'n6' is the 'seed object'. The items following will be made into entities with the same properties as the 'seed'.

14 Tools>Command Window. Type 'createcopies' or look for it in the drop down menu. A box appears saying "26 items created." <OK>.

15 Back in the home view, turn on Snake filter turn off Curve filter. Select Copy Snake 'n7'.

16 With Ctrl pressed, drag a box around all the Projected Snakes, which make up the text string. Remove the original snake 'n6' from this Selection Set.

17 Tools>Command Window 'createcopies'

18 All the snakes have been copied to the Revolution surface, effectively wrapping them about the surface.

Delete

Deletes all entities in the selection set, if such deletion is legal; i.e., if no entities in the selection set have dependents outside the selection set. If it can't delete the entire selection set, it does nothing. If "LastEdited" entity is in the selection set, this pointer is set to NULL. "LastSelected" pointer is set to NULL.

File.Close

Closes the current model, just like File/Close on the menu.

File.Export3D.3DA

File.Export3D.3DA [filename [.ext]]

Exports a 3DA wireframe file, just like File/Export3D/3DA, but without dialogs. If the selection set is empty, the file will have include all visible entities; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .3DA extension. If no extension is given, SurfaceWorks will use .3DA.

File.Export3D.3DG

File.Export3D.3DG [filename [.ext]]

Exports a 3DG FormZ triangle mesh file. If the selection set is empty, the file will include all visible surfaces and triangle meshes; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .3DG extension. If no extension is given, SurfaceWorks will use .3DG.

File.Export3D.GDF1 command

File.Export3D.GDF1 [filename[.ext] [tol [order]]]
with the wetted surfaces selected.

Exports a WAMIT higher-order GDF file of B-spline
surfaces (IGDEF=1).
tol has units of length; default = .0005 times model size.
(* can be used for default tolerance.)
order = B-spline order; default = 4 (cubic).
(order can be specified as 0, to use exact NURBS data.)
(Special WAMIT authorization is required.)

File.Export3D.NTL

File.Export3D.NTL [filename[.ext] [tol]]

Exports NTL (Patran Neutral File) file of the selection set (or all visible entities, if nothing is selected). The default extension is .NTL. The default tolerance is .0005 times the model size

File.Export3D.OBJ

File.Export3D.OBJ [filename [.ext] [tol]]

Exports an Alias Wavefront OBJ file. If the selection set is empty, the file will include all visible entities; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .OBJ extension. If no extension is given, SurfaceWorks will use .OBJ. tol is the tolerance; default tol is .0005 * size.

File.Export3D.PAN

File.Export3D.PAT

File.Export3D.PAT [filename [.ext]]

Exports a PAT patch file, just like File/Export3D/PAT, but without dialogs. If the selection set is empty, the file will include all visible surfaces; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .PAT extension. If no extension is given, SurfaceWorks will use .PAT.

File.Export3D.PNL

File.Export3D.PNL [filename [.ext]]

Exports a .PNL file. If the selection set is empty, the file will include all visible surfaces; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .PNL extension. If no extension is given, SurfaceWorks will use .PNL.

File.Export3D.PTN

File.Export3D.PTN [filename [.ext]]

Exports a PTN patch file. If the selection set is empty, the file will include all visible surfaces; otherwise, only the selected entities are output. Symmetry images can be turned on or off with the SetSymmetry command. If filename.ext is absent, SurfaceWorks will use the .MS2 filename, with .PTN extension. If no extension is given, MultiSurfSurfaceWorks will use .PTN.

File.Export3D.STL

File.Export3D.STL [filename[.ext] [ascii]]

Saves an STL (stereolithography) file.

If anything is selected, uses the Selection Set; else all visible surfaces and triangle meshes.

Use ascii = 1 to save in ASCII format; default is binary.

File.Export3D.STP

File.Export3D.UTP

File.Export3D.UTP [filename[.ext]]

Saves a .UTP (triangular panel) file.

If anything is selected, uses the Selection Set; else all visible surfaces and triangle meshes.

File.Export3D.WRL

File.Export3D.WRL [filename[.ext] [kind]]

Saves a VRML (Virtual Reality Modeling Language) file

If anything is selected, uses the Selection Set; else all visible surfaces and triangle meshes.

kind = 1 or 2 for VRML 1.0 or 2.0 format. (Default = 2.)

File.Open

File.Open filename [.ext]

Opens a model file. The model must be in the directory specified by a previous SetPath command, or a fully qualified path must be supplied in the command. If no extension is specified, .MS2 is assumed.

FixNurbsKnots

FixNURBKnots [places]

Makes minimal adjustment to knot spacing of NURBS surface, Light NURBSsurface, NURBS Curve or NURBS Snake to allow evaluation without error.

FreezeFit

[Available when the selection set contains exactly one object from the following list of entities:

B-spline Fitted Curve	NUB-spline Fitted Curve
B-spline Fitted Snake	NUB-spline Fitted Snake
B-splijne Fitted Surface	NUB-spline Fitted Surface

The common feature of these entities is that they all embody curve or surface fitting operations, using either uniform or non-uniform B-spline math.]

This command accesses the curve-fit or surface-fit results from the fitted entity, creating free-standing geometry (consisting of NURBS curves, snakes and / or surfaces, and the point entities needed to support them) that duplicates the fitted curve or surface. The term “freeze” implies that a support relationship is broken: the “frozen” curve or surface and its generated parents do not depend on the fitted curve or surface or their parents.

Applications of Freeze Fit

Simplifying a “heavy” NURBS surface

Frequently surfaces that arrive in IGES files have a totally obscene number of control points. This makes them slow to load and edit, and tedious or impossible to modify. Frequently you can obtain a very close BFit or NUBFit approximation with a modest number of control points, and then Freeze Fit to replace the heavy surface with a much lighter and more easily editable one.

Preparing geometry for IGES export

BFit and NUBFit approximations prior to IGES export allow management of the approximations on a surface-by surface basis. Freeze Fit can be part of this process. However, note it is usually just as effective to export the BFit or NUBFit surface directly – you get exactly the same results in the IGES file.

Access to NUBFit knots

The knot lists used in a NUBFit entity are stored internally and not directly accessible. Freeze Fit will turn them into Knot List entities, which can be used in NUBFitting adjacent surfaces to achieve accurate joins in the exported IGES. If all you want from a Freeze Fit operation is the Knot Lists, you can easily select and delete the other generated entities.

Replacing an expensive intersection or procedural object

IntSnakes generated by surface-surface intersections, and Procedural objects involving intersections and projections, are computationally intensive and may make your model slow and unresponsive. They are also vulnerable to radical failures caused by possibly small geometry changes. When the supporting geometry is in a final configuration, consider replacing that expensive snake or surface with a light, robust BSnake or NURBSurf, generated by fitting followed by Freeze Fit.

Breaking a connection between parts of a model

Suppose you have a complex hull and want to build a deck and superstructure on it. The hull is already under construction and isn't going to change. A natural first step is to make a snake on the hull for the deck edge, then everything beyond depends on that snake. But that means the whole complex hull model has to be part of your superstructure model. If you fit and freeze that snake as a curve, then build onward from the curve, you can isolated the superstructure in a separate model.

Breaking dependence on external files

Tabulated Curves and Tabulated Surfaces supported by 3DA files are a good way to import curve and surface data, for example from a DXF file (via DXF2MSF utility). But if you build directly on the TabCurves and TabSurfs, your model will always depend on those files. If you fit and freeze the TabCurves and TabSurfs, you can delete them from your model and build on the standalone curves and surfaces that result. Moreover, you will be able to edit and modify the "frozen" entities, whereas the Tabulated entities can only be modified by changing the data in their files.

GetPointsAt

GetPointsAt [mask [layer layer …]] (with one point selected)

List points at the location of the selected point.

Mask is a string of subclass chars, e.g. "pm"* admits all points.

Guests

Displays a message box which shows all the guests of the selected entity (if any). ("Guests" is the inverse relationship of 'host'.)

Help, or Help *command*

With no argument, displays a dialog box listing all the documented commands, in alphabetical order.

Help with a command argument (e.g. "Help File.Open") displays a dialog box that explains the syntax for the specified command.

Host

Displays a message box which identifies the host(s) of the selected entitity.

Import3DA

Import 3DA filename[.ext] [kind [layers]]

Imports 3DA file into points and / or curves.
Kind = 0, points only
Kind = 1, Type-1 BCurves
Kind = 2, Type-3 CCurves

Import3DG

Import3DG [filename[.ext]]

Imports a FormZ 3DG triangle file, creating one or more Triangle Mesh entities from the data.

ImportGDF1 command

ImportGDF1 filename[.ext] [kind]

Imports a WAMIT higher-order GDF file
in IGDEF = 1 format (B-spline surfaces).
kind = 0 (default) for light NURBS surfaces;
kind = 1 for points and NURBS surfaces.

ImportNTL

ImportNTL filename[.ext] [kind [layer]]

Imports geometric data from a Patran Neutral file into the current model. kind is inactive.

ImportOBJ

ImportOBJ [filename[.ext]]

Imports triangular facets from an Alias WaveFront OBJ file, creating one or more Triangle Mesh entities from the data.

ImportOFE

ImportOFE filename[.ext] [kind [layer]]

Imports .OFE offsets file as type-1 BCurves.

kind = 0, all on one layer (default);

kind = 1, each station on a separate layer.

default layer = 0.

With this command, in conjuntion with WriteOFE, SurfaceWorks can be used as an OFE Editor.

ImportPLX

ImportPLX [filename[.ext]]

Imports a PLX (AeroHydro plate expansion) file, creating one or more Triangle Mesh entities from the data.

ImportRAW

ImportRAW [filename[.ext]]

Imports a POV-ray raw triangle file, creating one or more Triangle Mesh entities from the data.

ImportSTL

ImportSTL [filename[.ext]]

Imports an STL (stereolithography) file, creating one or more Triangle Mesh entities from the data

ImportTable

ImportTable filename[.ext] [kind [layer]]

Imports table file into points and/or curves. If no extension is given, .TXT is used.

Kind = 0, points only

Kind = 1, Type-1 BCurves

Kind = 2, Type-3 CCurves

ImportUTP

ImportUTP [filename[.ext]]

Imports a UTP (unordered triangular panels) file, creating one or more Triangle Mesh entities from the data.

Insert

Insert type_name

Creates a new entity of the type specified by type_name. (type-name is NOT case sensitive.)

Selection set can contain preselected parents.

There are many more RGKernel Entities than are on the menu and fully documented. The Insert Command gives you access to quite a few more tools. For a number of reasons thay have been left off the menu, ranging from obscure usage to lack of robustness. Many entities are familiar to long time SurfaceWorks users and we want to make sure they are accessible to them and any other advanced users. A list and short description follows. The name in **bold** is the RGKernel name which should be used in the command syntax.

Example: Insert breakbead

Breakbead A bead located at a breakpoint of a curve.

Breakring A ring located at a breakpoint of a snake.

IntSnake2 Alternate intersection method. Should be used when menu driven Intersection Snake give poor results.

PolyCurve The original version of PolyCurve and required by some users for certain applications. The biggest difference is the ability to set the location of the ending t-values of the various segments of the PolyCurve.

PolySnake Snake version of the PolyCurve explained above.

PolySurf Surface version of the PolyCurve explained above. This is an entity which can cause more problems than it solves. The biggest thing to look for is the introduction of breaklines into the model which could have detrimental effects further down the dependancy tree.

TMTrim1 Trimmed TriMesh

LabelTrimesh

LabelTriMesh [code]

with one Trimesh-class entity selected.
Creates TM magnets labeling nodes, links, and / or triangles with their numbers.
code is bitwise: 1 for nodes + 2 for links + 4 for triangles.

MCReversal

MCReversal [on|off]

Changes the way surfaces are created in order to correct the twist caused by conflicting orientation of supporting curves. It does not actually change the orientation of supporting entities. If no parameter is supplied, “on” is assumed; the system default is also “on”. Turning the command off allows the creation of a twisted surface in the event that this is what the user had in mind.

NOTE: Be aware that if you use this command to repair a twisted surface created with MCReversal turned off, the surface will not be corrected when the command is entered and MCReversal turned on. In order to correct the surface you must edit the attributes for the curve.

MakeTriMesh

MakeTriMesh [light]

With 1 surface OR with at least 3 lines selected.

Makes a Triangle Mesh

(light = 0, default) or a Light Triangle Mesh (when light = 1).

NUBFitSurf Command

Nonuniform B-spline fitting of a surface, taking breaklines into account.

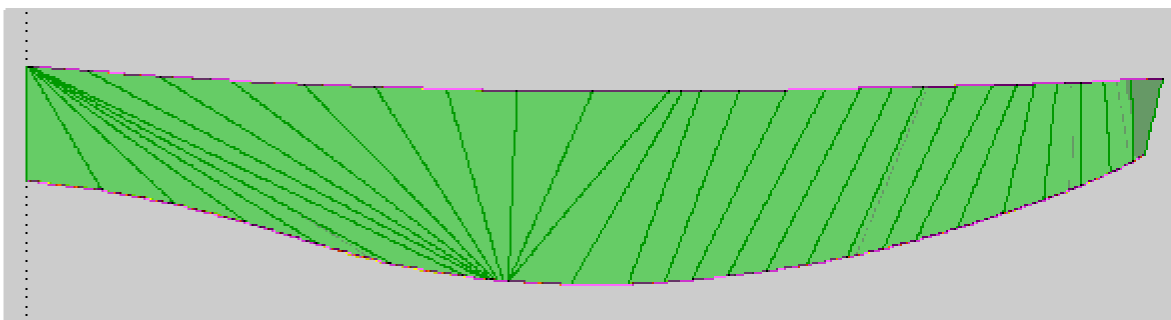
The surface is broken along breaklines into "panes". The “panes” are fitted with compatible NUBS surfaces, and are (optionally) reassembled into a single NURBS surface with multiple knots as needed at the breaklines. It has been our experience, if the surface is intended for export, the surface should be left as separate panes. In most cases these panels can be re-assembled in the destination CAD package.

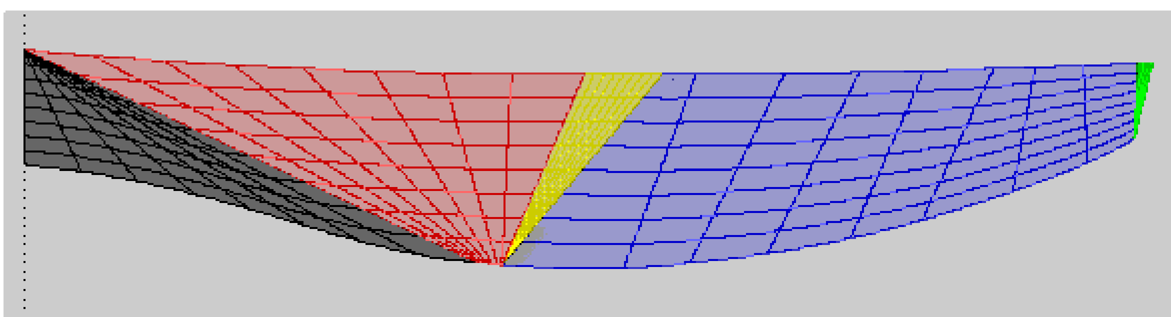
In SolidWorks: Insert / Surface / Knit

One big caveat: These new surfaces DO NOT update with the parent surface.

Examples

The top figure shows a typical developable side panel for a metal boat hull, which would be very hard to approximate in an IGES export or a transfer to SolidWorks. In the lower figure, the panel has been divided into 5 separate surfaces. The command is detailed below the images.





The Command

NUBSurfFit[utype [nulim [vtype [nvlim [tol [assemble [lt]]]]]]]

utype (default 3 = cubic) The degree of B-Spline used to fit the surface in the u-direction

nulim" The max number of control points in the u-direction

vtype (default 3 = cubic) **The degree of B-Spline used to fit the surface in the v-direction**

nvlim **The max number of control points in the v-direction**

tol tolerance (default = 1e-5 x surface size)

assemble (0|1) into single NURBS surface (default = 1)

lt Surface type: 1 for LtNURBSurf (default =1)

The command (Tools/Command Window) used to fit the above surface in 5 panes is:

NUBSurfFit 3 32 1 2 .001 0 1

NUBSurfit - Fit the selected surface using a degree 3 spline in the u-direction and a max of 32 control points. Fit the v-direction with a degree 1 spline with 2 control points. The tolerance will be 1E-3. The surface will be broken into separate panes, which will be LtNurbs Surfaces.

PrintBevels

PrintBevels [filename[.ext] [ds]]

With 1 surface plus 1 or more X=constant contour entities selected.

Writes a file of bevel angles with respect to X-direction.

default extension = .TXT

ds = arc length interval between points (default 1.)

PrintSS

PrintSS [filename[.ext]]

Prints the names of the selected entities to a file.

RealValues

RealValues command reveals the current values and unit dimensions of real-valued entities (Variables and Formulas) in your model.

Relationship

The Relationship command reveals the dependency relationship (if any) between two selected entities. It is useful for tracking dependencies in a complex model, especially one that you're unfamiliar with, or have forgotten the details of.

Example: Open Demo.ms2; select XContours 'stations' and AbsPoint 'P22'. Window / Command and issue the command:

Relationship

(This command has no parameters or modifiers.) You get a message box with this information:

> XContours stations is a 3rd generation descendant of AbsPoint P22
>
> AbsPoint P22
>
> BCurve MC2
>
> CLoftSurf hull
>
> XContours stations
>
> File saved: c:\%Relation.TXT

If you look up the referenced file, you'll find it contains other useful information to help you find and identify the intermediate entities:

```
DEMO  23-Jun-2004  17:08:01
color vis layer  name
  14    1    0  AbsPoint P22
  11    1    0  BCurve MC2
  10    1    0  CLoftSurf hull
  12    1    0  XContours stations
```

Note that the path between two entities is not necessarily unique (as it is in this example). Relationship displays the first path it finds connecting the two entities.

Rename

Rename [root [base]]

Renames the selection set. root = base name; use * for automatic root. base = starting number.

For example, with four points selected, the command

RenameEntities `P 5`

will attempt to rename the points as 'P5', 'P6', 'P7', 'P8'. If any of these names are unavailable (i.e, already in use by an entity that is not currently selected), no names are changed.

ScaleSubDivs

ScaleSubDivs [scale factor]

Multiplies the sub-divisions of curves and surfaces, in the model, by the scale factor. This command is commonly used to lessen the sub-divisions in a model to improve performance.

Select

Select [name name ....]

This clears the working selection set and selects the named entity(s).

Select+

Select+ [name name ....]

This adds the named entity(s) to the working selection set, without clearing the set.

Select-

Select- [name name ....]

This removes the named entity(s) from the selection set.

SelectForComponent

SelectForComponent is a convenience for collecting together the set of entities to be included in a component (MC2) file. This is especially convenient when the component is a complex one with many entities, or when you are refining the component and need to repeatedly save revised and improved versions.

To use the command, first make two EntitityLists:

(1) a list of the supports needed for the component (we'll call this EntitityList 'supports') and

(2) a list of "products" -- the essential entities that the component needs to produce (we'll call this EntitityList 'products').

Then select 'supports' and 'products', in this order (the order is important), Window / Command and give the command:

SelectForComponents

(This command has no parameters or modifiers.) Following the command, the selection set should contain exactly the set of entities that are either products or ancestors of products and are descendants of the "supports". Then you're all set for File / Component / Save.

SetDivMult

SetDivMult [value]

Sets the divisions multiplier for the current model. value is screened to be > 1 and < MAX_DIVMULT; no change is made if value is out of bounds. If value is omitted, divmult is set to 1.

SetFloat

SetFloat entity index value [0]

‘entity’ is an entity name. This command sets a new value for one floating point number in the entity data, the one denoted by index. If the optional 0 is appended to the command, no update is performed following the data change. Otherwise, the entity and all its descendants (to any generation) are updated, and the change is registered with the Undo system.

You can figure out how many floating point numbers an entity has in its data, and the order in which they occur, by looking up its Entity Specification. For example, a Frame Point has three floats: its X, Y and Z coordinates, in that order. To set Z of P11 to 3.85, you would give the command SetFloat P11 3 3.85. A RevSurf has two floats: angle1 and angle2, in that order. To set angle1 of RevSurf body to -90, you would give the command SetFloat body 1 -90.

SetInteger

SetInteger entity index value [0]

‘entitity’ is an entity name. This command sets a new value for one integer in the entity data, the one denoted by index. If the optional 0 is appended to the command, no update is performed following the data change. Otherwise, the entity and all its dependents (to any generation), the model is updated, and the change is registered with the Undo system.

You can figure out how many integers an entity has in its data, and the order in which they occur, by looking up its Entity Specification. Note that:

> the integers in all curves and snakes start with 2 integers for t-divs and t-subd; the integers in all surfaces start with 5 integers for u-divs, u-subd, v-divs, v-subd and orientation.

the integers in all solids start with 7 integers for u-divs, u-subd, v-divs, v-subd, w-divs, w-subd, and orientation.

For example, a BCurve has three integers: t-divs, t-subd, and type, in that order. To set t-subd of MC1 to 4, you would give the command:

SetInteger MC1 2 4

A CLoftSurf has six integers: u-divs, u-subd, v-divs, v-subd, orientation and type. To set type of CLoftSurf hull to 3, you would give the command:

SetInteger hull 6 3.

SetPath

SetPath path

This establishes a directory for subsequent File commands. If you don't set a path, SurfaceWorks will use the root directory.

SetSymmetry

SetSymmetry [on | off]

Sets symmetry images on / off, for export commands.

If no argument, sets symmetry ON.

SmoothWireframe

SmoothWireframe [1|0]

Toggles Smooth Wireframe on and off.

Solve

Solves an N-Degrees Of Freedom geometric problem stated by the Selection Set.

SelSet: free points with N Degrees of Freedom, plus N [point, plane / surface] pairs.

Strain

Strain(Surface / TriMesh, index)

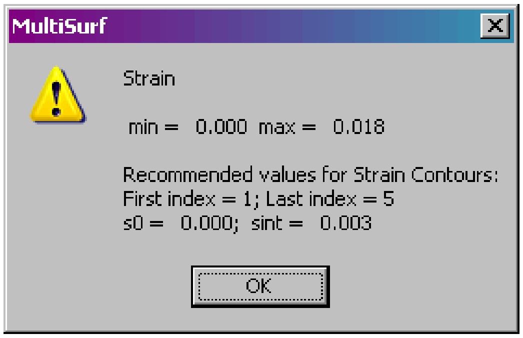
Surface / TriMesh is a surface or TriMesh entity

index = 0 or 1, for minimum or maximum strain

This function reports the strain range for an Expanded Surface or Expanded TriMesh. (It accepts any surface or TriMesh entity, but just returns 0 if the entity is not one of these two types.)

With one Expanded Surface or Expanded Triangle Mesh selected. Shows the range of strains required for flattening, and recommended settings for Strain Contours.

Example output:



TempCopy

TempCopy [color] -- Creates a temporary Wireframe entity from the selection set. At least one visible entity must be selected. 'color' is a color code, 0-255.

Note: Default color is 7 (light gray), which will probably be invisible if you use a light gray background.

TestEval

TestEval [N]

Re-evaluates the selected entity N times and reports the time taken.

Toposort

Sorts the model into a top-down or "topological" order, in which any parent is listed before any of its childen. This makes a model file much easier to read. (Theoretically, model evaluation is independent of the order the entities are listed in, but there have been occasional bugs that have broken this rule.)

WFCurveFit

WFCurveFit [nv [type]]

Fits Bcurves to polylines of selected WireFrame entitity.

Defaults: nv=6; type = 3

WriteOFE

WriteOFE filename[.ext]

With a set of transverse type-1 B-Spline Curves selected.

Writes an offset file; default extension is .OFE.

With this command, in conjuntion with ImportOFE, SurfaceWorks can be used as an OFE Editor.

Using SurfaceWorks with SolidWorks

Introduction

SurfaceWorks adds to SolidWorks the capability to create a wide variety of freeform and complex curves and surfaces, while at the same time maintaining bi-directional associativity and relational editability. SurfaceWorks encourages the designer to create and maintain rich relationships between SolidWorks and SurfaceWorks geometry that express and capture design intent.

If a MultiSurf user desires to incorporate SolidWorks Integration into their MultiSurf License, it will be converted to SurfaceWorks with a marine option. The following chapter, with all the SurfaceWorks references, will now apply.

This chapter deals with using SurfaceWorks in SolidWorks Integration mode. Since the details of using SurfaceWorks once you are in SurfaceWorks is handled in the body of this User's Guide, this chapter deals mostly with the interface between the two programs.

Entity Dependencies

In both SurfaceWorks and SolidWorks, you can establish relationships between entities. SurfaceWorks supports numerous kinds of relationships, such as (but not limited to):

linear distance
arc-length distance
coincident with (or constrained to) a curve
coincident with (or constrained to) a surface
mirrored location
projected location
arc tangency
G1 and G2 surface continuity
...

The relationships are stored in the .sldprt, so that if one or more of the underlying entities is changed, the dependent entity (and all of its dependents, and all of their dependents, etc.) automatically change and update to preserve the relationships.

Built-in Relationships

SurfaceWorks and SolidWorks differ a bit in how relationships are created. In SolidWorks, an entity is created and relationships are added afterwards. Automatic relations are the exception.

In SurfaceWorks, relationships are built into the entity definitions. For example, where in SolidWorks you would create a point, then add the relation to be coincident with a particular curve, in SurfaceWorks you just create a Bead entity, which is, by definition, coincident with its parent curve. This makes for a larger number of entity types to choose from, but each type of entity has a descriptive name that is easy to learn. For example, there are several types of bead entity, including the Bead entity just introduced (which is a bead located by t-parameter), the Arc-length Bead (a bead located by arc-length), and the XYZ Bead (a bead located by an X,Y, or Z value). For details about these entities, see their respective entries in the “Entity Descriptions” chapter.

SurfaceWorks also has its own version of automatic relations. For example, you alternatively can create a Bead entity by preselecting a curve and clicking a “Quick Point” — a bead will be automatically created on that curve. For details, see “Creating Entities - Quick Points and Quick Splines - Quick Points.”

SolidWorks Interface Changes When SurfaceWorks Is Added In

When you Add-In SurfaceWorks, the following changes occur in your SolidWorks program window:

- A SurfaceWorks menu is added to the SolidWorks menu bar when a part is opened. It includes options to invoke the Surface Modeler and to manage the relationships between surfaces in SurfaceWorks and SolidWorks. The menu choices are:

  Modeler
  Add SurfaceWorks Parent
  Remove SurfaceWorks Parent
  Manage SurfaceWorks Parents

- SurfaceWorks Help and About options are added to the SolidWorks Help menu.

Beginning Your SurfaceWorks Work

You can begin your SurfaceWorks work in two ways:

- By opening a new part and going immediately from SolidWorks to SurfaceWorks.
- By specifying SolidWorks parents for SurfaceWorks geometry.

Beginning a SurfaceWorks Project “From Scratch”

To begin “from scratch” in SurfaceWorks:

1. In SolidWorks, **open a new Part**.
2. From the SolidWorks menu bar, choose **SurfaceWorks>Modeler** to start SurfaceWorks.

What SolidWorks Geometry Can Be Used as SurfaceWorks Parents

Currently, you can use the following kinds of SolidWorks geometry as SurfaceWorks parents:

- 3D and 2D Sketches
- Reference planes
- Reference curves (3D Spline Curves, Curves from File, Helix Curves, Imported Curves, Projected Curves, and Composite Curves)
- Edges
- Planar and non-planar faces
- SolidWorks surfaces

Note: Sketches, reference planes, and reference curves are features, while edges and faces are not. Hence there are two different ways to work with them in the software.

Specifying SolidWorks Geometry as the Basis for SurfaceWorks Geometry

You can assign SolidWorks geometry as SurfaceWorks parents:

- By selecting individual or groups of sketches, features, edges, and planar faces via the FeatureManager and / or the graphic view.

Assigning Parents Individually or by User-Specified Groups

To specify SolidWorks parents for SurfaceWorks geometry:

1 In SolidWorks, **open an existing Part** or open a new Part and create the geometry you want to use as SurfaceWorks parents.

2 In the FeatureManager (for planes, sketches, reference curves, and user-named edges or faces) or in the graphic view (for planes, edges, and faces; not for sketches or reference curves):

- Use the **left mouse button** to select the entity, then choose **SurfaceWorks>Add SurfaceWorks Parent**

  or
- Use the **right mouse button** to select the entity and bring up the shortcut menu, then choose **Add SurfaceWorks Parent**.

3 From the SolidWorks menu bar, choose **SurfaceWorks>Modeler** to start SurfaceWorks and use the SolidWorks parent geometry.

Note: The Add and Remove SurfaceWorks Parents commands work as a toggle — that is, if the selected entity is of an exportable type AND has not yet been chosen as a parent, Add SurfaceWorks Parent is enabled, and if the entity has already been chosen as a parent, Remove SurfaceWorks Parent is enabled.

Managing SurfaceWorks Parents

The Manage SurfaceWorks Parents dialog, accessed from SurfaceWorks menu, serves this purpose:

- The Exports - Imports tab includes information such as which SolidWorks entities have been assigned as SurfaceWorks parents and which SurfaceWorks entities correspond to what SolidWorks imported geometry (reference surfaces and imported solids). You also can use this tab to remove SurfaceWorks parents and to select entities.

The Exports - Imports Tab

The Exports - Imports tab displays two synchronized lists:

- The **Exports** column lists the SolidWorks entities that have been selected and added as parents in SolidWorks. Additionally, the origin icon indicates a SurfaceWorks import (in Imports column) that is based entirely on SurfaceWorks geometry (i.e. has no SolidWorks parents).

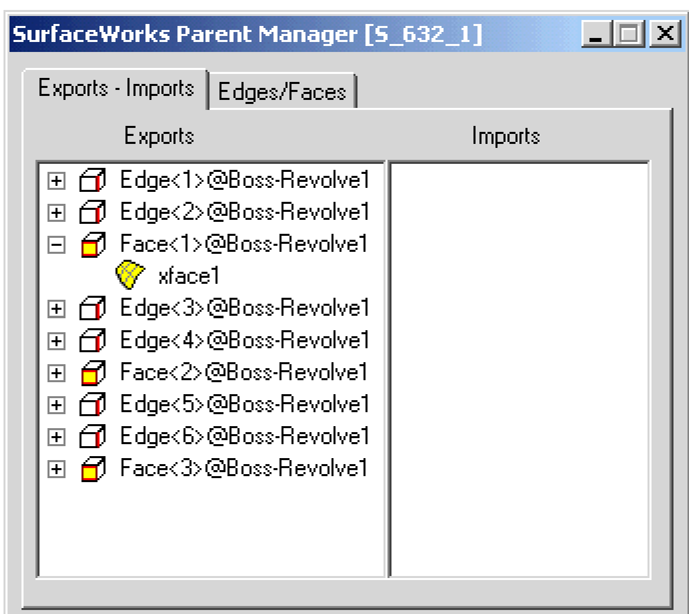
- The **Imports** column lists the entities transferred from SurfaceWorks to SolidWorks.

As in the FeatureManager, ⊞ and ⊟ expand and contract the list information:

- An **Exports** entry expands to show the list of SurfaceWorks entities into which the SolidWorks entity has been translated.

- An **Import** entry shows the SurfaceWorks surface(s) the SolidWorks reference surface or solid comes from.

When you select a parent listed in the Exports column, the edge or face it corresponds to will highlight in the graphic display.



Using the Exports - Imports Tab to Remove SurfaceWorks Parents

You also can use the Export - Import tab to remove SolidWorks entities from SurfaceWorks parenthood, in either of two ways:

- In the Exports column, right-click the entity you want to remove, then click Remove SurfaceWorks Parent from the popup.
- In the Exports column, select (highlight) one or more entities to remove, then choose SurfaceWorks>Remove SurfaceWorks Parent from the menus.

The Edges/Faces Tab

The Edges/Faces tab has two check boxes ("All Edges" and "All Planar Faces") and two buttons ("Add" and "Remove") which let you add or remove all edges and/or all planar faces as SurfaceWorks parents. Should you desire, after adding by the "all" method of the Edges/Faces tab, you can use the Exports - Imports tab to selectively remove one or more edges/faces from SurfaceWorks parenthood.

Note: When you add a planar face as a parent (either individually or by the "all" method), its edges automatically are added as well. But if you then remove a planar face from parenthood (by either method), the edges are NOT automatically removed.

SolidWorks Geometry in SurfaceWorks

When SolidWorks geometry is transferred into SurfaceWorks, the SolidWorks entities are translated into corresponding SurfaceWorks entities. Oftentimes, additional intermediate entities are created by SurfaceWorks in order to specify the final entity and/or to establish proper relationships between entities. For example:

- When a 2D sketched circle is translated into the 3D entity of SurfaceWorks, an additional point is needed to identify the plane of the circle.
- When a SolidWorks face is translated into a SurfaceWorks Trimmed Surface, a basis surface must be created as well.

Defaults for SolidWorks Entities in SurfaceWorks

- All transferred SolidWorks entities are given a unique name that begins with the letter "x"; e.g. 'xpt1' (a 3D point), 'xarc5' (an arc), 'xcv1' (a curve), 'xedge2D~4' (a 2D edge), 'xface2' (a face), 'xplane7' (a plane).
- All transferred SolidWorks entities come in colored dark magenta.
- All transferred SolidWorks entities are put on Layer 0, which is named SolidWorks Entities.
- Upon initial transfer, all points are hidden except for curve endpoints and points with no children. Should you need to display any of the hidden points, you have a number of methods available to you for doing so:

- Right click on the entity(s) you want to display in the Surface Manager or Available Entities pane and choose **Show** from the menu.
- Select the entity(s) you want to display in the Surface Manager or Available Entities pane and then click or choose **Show-Hide>Show Selection Set.**
- To display <u>all</u> points, click or choose **Show-Hide>Show Points**.

- Upon initial transfer, additional geometry created to fully support SolidWorks entities in SurfaceWorks (e.g. the basis surface for a Trimmed Surface) is hidden. Generally, you will have no use for this geometry, but if you click or **Show-Hide>Show All**, these entities will display.

Note: The default current layer when you first enter SurfaceWorks is 1 (not 0), so inserted SurfaceWorks entities automatically go on a layer different from that of the SolidWorks geometry.

Sketch Entities that Do Not Transfer to SurfaceWorks

The following sketch entities do not transfer to SurfaceWorks:

- Closed splines — that is, splines that have the same start point as end point. This includes offsets of an ellipse (an ellipse itself transfers fine).

What You Can Do with SolidWorks Entities in SurfaceWorks

In SurfaceWorks, you can:

- View the transferred entities
- Select translated entities as parents for SurfaceWorks entities
- Select translated entities for **Show-Hide** options (e.g. **Show-Hide>Hide Selection Set**)
- Hide or display the translated entities by turning off / on Layer 0 (**Tools>Layers**)

In SurfaceWorks, you <u>cannot</u> modify translated entities. To edit these entities, return to SolidWorks (**File>Exit & Return to ________.SLDPRT**).

SurfaceWorks Geometry in SolidWorks

When you return to SolidWorks after having created SurfaceWorks geometry, only the SurfaceWorks surfaces are transferred into the SolidWorks model:

When these surfaces are <u>not</u> stitched together to enclose a volume, they become SolidWorks reference surfaces. In the FeatureManager, reference surfaces are identified by the icon.

When the SurfaceWorks surfaces are stitched together to enclose a volume, they are transferred to SolidWorks as imported solids. In the FeatureManager, imported solids are identified by the icon.

You can use the SurfaceWorks-generated reference surfaces just as you would reference surfaces created in SolidWorks, e.g. you can:

- Cut with them (**Insert>Cut>With Surface**)
- Extrude up to them (**Insert>Cut>Extrude>Up to Surface** or **Offset from Surface**)
- Thicken them (**Insert>Base>Thicken**)

Non-surface SurfaceWorks entities (e.g. curves and points) are not transferred into SolidWorks as useable geometry, but they are saved into the .SLDPRT file — SolidWorks **File>Save** and **File>Save As** save into the Part file all the SurfaceWorks model data necessary to support SurfaceWorks-SolidWorks associativity.

Removing SolidWorks Geometry from Use as SurfaceWorks Parents

There may be occasions when you want to remove SolidWorks geometry from use as SurfaceWorks parents.

Note: If the SolidWorks geometry you remove from SurfaceWorks parenthood has SurfaceWorks children, you will be offered the options to orphan those SurfaceWorks children or to delete them from the model. You will have the opportunity to cancel the removal of parenthood.

To remove SolidWorks geometry from use as SurfaceWorks parents:

1 In the graphic view, in the FeatureManager (this latter applies only to sketches, planes, reference curves, and user-named edges or faces), or on the Exports-Imports tab of the Manage SurfaceWorks Parents dialog:

- Use the **left mouse button** to select the applicable entity, then choose **SurfaceWorks>Remove SurfaceWorks Parent** from the menus

  or
- Use the **right mouse button** to select the applicable entity and bring up the shortcut menu, then choose **Remove SurfaceWorks Parent**.

OR

On the Edges / Faces tab of the Manage SurfaceWorks Parents dialog, **check the box for All Edges and/or All Planar Faces, click the Remove button**, then click **OK** to the dialog.

2 If a SolidWorks parent has SurfaceWorks dependents, you will be warned that removing the feature will orphan the SurfaceWorks dependents and you will have two options. Choose one by clicking its radio button:

Keep Dependents — the dependent SurfaceWorks geometry will be retained in the model but it will have no connection to its former SolidWorks parents, so any associativity will be gone

Delete Dependents — the dependent SurfaceWorks geometry will be deleted from the model

3 Click **OK** to continue with SolidWorks parent removal and SurfaceWorks dependent orphaning or deletion; click **Cancel** to abort and return to SolidWorks with parent assignments unchanged.

Note: When you remove edges or faces, the numbering of subsequent edges or faces listed in the Manage SurfaceWorks Parents dialog will change. In SurfaceWorks, a name is "attached" to a particular entity (you can edit it, but it won't change otherwise); in SolidWorks, edge and face enumeration is not "attached" to a particular edge or face.

Deleting SolidWorks Geometry that Has SurfaceWorks Dependents

If you choose to delete SolidWorks geometry that has SurfaceWorks dependents:

1 You will get the standard SolidWorks request that you confirm deletion of the feature and its dependent features, including reference surfaces or imported solids transferred in from SurfaceWorks, will be deleted.

Choose Yes, No, or Cancel.

2 Then you will get the dialog that lets you choose to either:

Keep Dependents — the dependent SurfaceWorks geometry will be retained in the model but it will have no SolidWorks parents, so any associativity will be gone

Delete Dependents — the dependent SurfaceWorks geometry will be deleted from the model

Choose one or the other option; OK. Note that since you have already confirmed deletion of the SolidWorks geometry, you must choose one of the SurfaceWorks options here and you have no option to Cancel.

Stitching SurfaceWorks Surfaces Together for SolidWorks

In SurfaceWorks, the accurate and durable way to join adjacent surfaces is to make them share a common edge curve/snake. If you want such surfaces stitched together in SolidWorks, to become either a single Reference Surface or an Imported Solid, you use SurfaceWorks' stitching function in the Entity Transfer dialog ( or **Tools>Entity Transfer**).

In the dialog, check the boxes for the surfaces you wish to transfer and which of those you wish to stitch (and/or use the appropriate transfer and stitch buttons). Upon return to SolidWorks, an open (not enclosing a volume) set of stitched surfaces will become a single Reference Surface; a set of stitched surfaces that encloses a volume will become an Imported Solid.

Making a Hole with Sides Normal to a Surface (Using a SurfaceWorks Surface to Cut a Hole)

Sometimes you may want to cut a hole in a solid and have the resulting inner faces of the hole remain normal to the outer face of the object. You can do this by creating a Trimmed Surface with the hole in it. When you take this surface back into SolidWorks, you can thicken it into a solid body and the inner face of the resulting hole will remain normal to the front and back faces of the body.

Basic steps:

1. Create the basis surface.
2. Make a snake or snakes outlining the hole you want to create.
3. Create Edge Snakes that completely bound the basis surface.
4. Select the inner boundary snakes (in sequential order) and then the outer boundary snakes (in sequential order).
5. Create a Trimmed Surface using the above parents.
6. Transfer the Trimmed Surface back to SolidWorks and thicken.

Editing Models

What You Edit Where

The rules for editing when using the SolidWorks-SurfaceWorks combination are simple:

Geometry you create in SolidWorks you edit in SolidWorks.

Geometry you create in SurfaceWorks you edit in SurfaceWorks.

Of course, when you edit an entity in one or the other program, the changes are propagated throughout the model, so a direct edit in SolidWorks may affect SurfaceWorks (as well as SolidWorks) geometry and vice versa. For example, imagine a model in which: a SolidWorks thickened base feature is made from a reference surface based on a SurfaceWorks B-spline Lofted Surface that is built from three SolidWorks splines and one SurfaceWorks B-spline Curve, which in turn is based on one Point with absolute coordinates and several Points with relative positions. Here’s a summary stack of the dependencies involved:

Thickened base feature (SolidWorks)
    Reference surface (SolidWorks)
        B-spline Lofted Surface (SurfaceWorks)
            3 splines (SolidWorks) + a B-spline Curve (SurfaceWorks)
                Points + Points with Point parents (SurfaceWorks)

If we change the shape of any one of the SolidWorks splines, the shape of the SurfaceWorks B-spline Lofted Surface will change, and therefore the SolidWorks reference surface and thickened base feature will change.

Alternatively, if we change the location of the SurfaceWorks Point, its Point children will move, the B-spline Curve will change location, and the shape of the B-spline Lofted Surface and the SolidWorks reference surface will change.

Editing in SolidWorks

Overview

Editing a model in SolidWorks can include modification or deletion of entities as well as creation of new sketches, planes, and features. You also can add or remove sketches and planes as SurfaceWorks parents. If you delete a SolidWorks feature that has SurfaceWorks children, you will be given the option to keep (as orphans) or delete the SurfaceWorks children.

If you make new planes, sketches, or features that you want to use as SurfaceWorks parents, be sure to add them as SurfaceWorks parents (**SurfaceWorks>Add SurfaceWorks Parents**) before you go back into SurfaceWorks.

Editing a Sketch That is a SurfaceWorks Parent

Here are a few notes about editing a sketch that already is a SurfaceWorks parent:

- Added sketch entities automatically will be transferred to SurfaceWorks when you next invoke the modeler.

- Deleting sketch entities usually will orphan the corresponding entities in SurfaceWorks, and any child SurfaceWorks entities may get moved in an unpredictable way.

- Changing an ellipse so that its major and minor axes are swapped will reorient by 90 degrees the corresponding ellipse in SurfaceWorks. This can give a SurfaceWorks surface built off it an unexpected twist.

Editing a Feature Whose Edges/Faces Are SurfaceWorks Parents

In SolidWorks, you can edit features whose edges and / or faces are SurfaceWorks parents and the corresponding SurfaceWorks entities will update accordingly. The same applies to adding child features to features whose edges and / or faces are SurfaceWorks parents — but keep in mind the following caveats:

- If editing the feature results in less edges / faces than before, then the SurfaceWorks entities corresponding to the no-longer-existing (consumed) edges / faces will be orphaned, that is, they will no longer have any SolidWorks associativity. This typically can happen when editing the parent sketch of a feature.

- If adding a child feature such as a fillet, or a cut on an extrusion, etc. that entirely consumes one or more edges / faces, or replaces edges with new edges in the same place, the corresponding SurfaceWorks entities will no longer automatically update, even if the feature gets updated. In such cases, you can enable SurfaceWorks to update by rolling the model back to the pre-consumption state. Then rolling the model back down will rebuild the part in an up-to-date state.

- If you want to edit a model (e.g. redimension it) that includes a cut feature that has the same edges as the imported surface that cuts it, you need to follow this procedure:

  1 Before you make the edit, roll up the model to above the cut feature.

  2 Redimension (or whatever).

  3 Rebuild the model.

  4 Roll back down.

The consumed state of a SurfaceWorks parent is marked in the Manage SurfaceWorks Parents dialog by a .

SolidWorks Sketches as Parents

In many cases, you will use SolidWorks sketches as parents for SurfaceWorks surfaces. A little planning will help make the transition between the two programs a smooth one.

Lofting

- Plan ahead for creating mirrored images of your surfaces where applicable. This means lofting of boundary curves should end at the plane or axis you plan to mirror across or around.

- If you only need a quarter of an ellipse or circle, it's quicker to trim them in SolidWorks than to create SubCurves in SurfaceWorks.

Otherwise, sketch as you would normally. If something unexpected comes up, there are many tools in SurfaceWorks to easily turn your sketches into proper SurfaceWorks parents.

Planar Faces

- Any planar face that will have SurfaceWorks geometry attached to it can simply be imported and SurfaceWorks geometry added. In SurfaceWorks, you will find the faces represented as Trimmed Surfaces.

- Should you want to add an adjacent tangent surface, you will need to use one or more of the trimming curves (Projected Snakes) as parents for a Tangent Boundary Surface.

Reference Guide to SurfaceWorks for SolidWorks Functions in SurfaceWorks

Entity Transfer

[Only available when the model contains at least one SurfaceWorks surface.]

Tools>Entity Transfer

Used to specify which SurfaceWorks surfaces to transfer to SolidWorks and which of those surfaces are to be stitched together to form a single SolidWorks Reference Surface or an Imported Solid. Note that only surfaces with coincident edges will stitch.

To make transfer and stitching settings:

1. Click or choose **Tools>Entity Transfer**.
2. In the **Transfer** column, check all the surfaces you want transferred to SolidWorks (the initial default when a surface is created is to transfer, so if you haven't changed any Transfer checkboxes, they will all be checked). Or use the applicable **Transfer button**: <**Set All**>, <**Clear All**>, <**Add All Visible**>.
3. In the **Stitch** column, check the surfaces you want stitched, or use the applicable **Stitch button**: <**Set All**>, <**Clear All**>, <**Add All Visible**>.
4. Click **OK**.

Transfer. When SurfaceWorks transfers surfaces to SolidWorks, it sends them as spline surfaces (the only kind of surfaces that SolidWorks understands). Since SurfaceWorks surfaces are not limited to spline surfaces, any SurfaceWorks surface that is not a spline surface must first be fitted with a spline surface, and then that fitted surface is sent to SolidWorks. When SurfaceWorks fits a surface with a spline surface, it tries to create a spline surface which matches the parent surface to within a SurfaceWorks-specified tolerance.

On occasion, a SurfaceWorks surface may be so complex or so convoluted that it cannot be fit with a spline surface within the tolerance. In this case, examine the results in SolidWorks — the results may be quite satisfactory anyway. If they are not, you may opt to return to SurfaceWorks and create a B-spline Fitted Surface representation of your original surface. Since a Fitted Surface is a spline surface, what you are doing is taking an active (rather than a passive) role in the fitting process. This may give you just enough added control over the representation of the surface.

Stitching. When stitching is off (not checked) for a surface, it is returned to SolidWorks as an individual Reference Surface. In order to stitch, surfaces must have coincident edges — e.g. two surfaces that share a common control curve as their adjacent edge, or a surface and a second surface built off an Edge Snake of the first surface. When stitching is on (checked), SolidWorks attempts to stitch all checked surfaces into a single SolidWorks Reference Surface. When the checked surfaces enclose a volume and stitch successfully, the result is an Imported Solid in SolidWorks.

Note: When the SurfaceWorks portion of a model is to become an Imported Solid in SolidWorks, this Imported Solid must be the initial solid in the model.

Exit & Return to ______.SLDPRT

File>Exit & Return to ___.SLDPRT

To exit SurfaceWorks and return to your SolidWorks model:

Choose **File>Exit & Return to *partname*.SLDPRT**.

If you made any changes to the SurfaceWorks model, you will be prompted whether or not to save the changes. If you choose **Yes**, SurfaceWorks applies your changes and passes the updated surface model to SolidWorks. Note that this saves the model back to SolidWorks, but you must then save the SolidWorks model if you want the changes to be saved. If you choose **No**, no changes are applied to the surface model, and the solid model remains unchanged when you return to SolidWorks.

Export .MS2

File>Export 3D>MS2

Output of your surface model in .MS2 file format is provided for compatibility with AeroHydro's MultiSurf product or for SurfaceWorks without SolidWorks Integration. If you, or someone you wish to share your surface model with, has MultiSurf or SurfaceWorks without SolidWorks Integration, you can export an .MS2 from SurfaceWorks and read it with either of these programs.

To export an .MS2 file:

Choose **File>Export 3D>MS2**.

Import .MS2

File>Import>MS2

[Only available when the model is empty.]

Lets you import a .MS2 file into SurfaceWorks and transfer that geometry back into an empty SolidWorks part. All the entities that have SurfaceWorks counterparts work just as though you had created them in SurfaceWorks. Entities that are not part of the SurfaceWorks offerings are available as parent geometry, but they are not directly editable (see Note below).

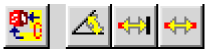
To import an .MS2 model into SurfaceWorks:

1. In SolidWorks, choose **File>New, Part**.
2. Choose **SurfaceWorks>Modeler**.
3. In SurfaceWorks, choose **File>Import>MS2** and **specify the .MS2 file** you want to bring in.

Note: Although there is no Edit dialog for some entities that can be imported into SurfaceWorks, these entities are relational entities with parents and you can change these entities in a more indirect manner. To change the shape, move the supporting points. To change properties such as layer, color, visibility, or divisions, do a multiple-entity edit — that is, select the entity twice or select two or more entities that share the property you want to change, then Edit Definition. Properties such as type or degree are not editable.

Toolbar Buttons

In SolidWorks Integration mode, the Entity Transfer/Stitching button, , is added to the SurfaceWorks **Tools** toolbar. The resulting toolbar looks like this:



Zoom In/Out

If you do not also have the Marine option:

- **Z** zooms out

- **Shift+Z** zooms in

Reference Guide to SurfaceWorks Functions in SolidWorks

Introduction

This chapter describes the SurfaceWorks additions to the SolidWorks menus and graphical interface. The functions are listed in alphabetical order, by function (e.g., About SurfaceWorks ... Remove SurfaceWorks Parent). Each function description includes:

- how it works and when it's available
- where to find it
- step-by-step instructions

About SurfaceWorks

Help>About SurfaceWorks

Displays version, build number and copyright information for SurfaceWorks. Choose **Help>About SurfaceWorks**.

Add SurfaceWorks Parent

SurfaceWorks>Add SurfaceWorks Parent

[Only available when the selected entity qualifies as a SurfaceWorks parent AND is not already assigned as a SurfaceWorks parent.]

Adds entities from the selected SolidWorks parent into the SurfaceWorks model for use in the SurfaceWorks **Modeler**. For a list of eligible parents, see "What SolidWorks Geometry Can Be Used as SurfaceWorks Parents" on page 563

Add a SurfaceWorks parent in any one of the following ways:

- In the FeatureManager, **select one or more eligible parents**, then choose **SurfaceWorks>Add SurfaceWorks Parent** from the SolidWorks menu.

- In the FeatureManager, **right-click the parent-to-be**, then choose **Add SurfaceWorks Parent** from the shortcut menu.

- In the graphic image, **right-click a plane**, then choose **Add SurfaceWorks Parent** from the right shortcut menu.

For further information on adding SolidWorks parents, see "Specifying SolidWorks Geometry as the Basis for SurfaceWorks Geometry" on page 563

Manage SurfaceWorks Parents

SurfaceWorks>Manage SurfaceWorks Parents

Provides SolidWorks-SurfaceWorks import/export information, lets you add and/or remove all planar faces and all edges as SurfaceWorks parents, and lets you remove individually any SurfaceWorks parents.

To see export/import information:

1 Choose **SurfaceWorks>Manage SurfaceWorks Parents**.

2 On the Exports-Imports tab, expand either list as necessary to see:

Exports. These are the SolidWorks entities which have been selected and added as parents in SolidWorks. Expanded entries list the SurfaceWorks entities into which the SolidWorks entity has been translated. Additionally the origin icon, , indicates a SurfaceWorks import (in Imports column) that is based entirely on SurfaceWorks geometry (i.e. has no SolidWorks parents).

Imports. These are the entities transferred from SurfaceWorks to SolidWorks. Expanded entries show the SurfaceWorks surface(s) the SolidWorks reference surface or solid comes from.

To add/remove all SolidWorks planar faces and/or edges as SurfaceWorks parents:

1 Choose **SurfaceWorks>Manage SurfaceWorks Parents**.

2 On the **Edges/Faces** tab, check the appropriate box, then click the **Add** or **Remove** button.

For information about other ways to add SolidWorks geometry as SurfaceWorks parents, see "Specifying SolidWorks Geometry as the Basis for SurfaceWorks Geometry" on page 563

To remove SolidWorks entities from SurfaceWorks parenthood singly or by user-specified groups:

1 Choose **SurfaceWorks>Manage SurfaceWorks Parents**.

2 On the **Exports-Imports tab, select one or more entities to remove**, then choose **SurfaceWorks>Remove SurfaceWorks Parent** (or you can right-click an individual entity and choose Remove SurfaceWorks Parent from the shortcut menu).

For information about other ways to remove SolidWorks geometry from SurfaceWorks parenthood, see "Removing SolidWorks Geometry from Use as SurfaceWorks Parents" on page 567

Modeler

SurfaceWorks>Modeler

[Only available when a SolidWorks part document is open.]

To invoke the SurfaceWorks Modeler:

Choose **SurfaceWorks>Modeler**.

This opens the SurfaceWorks surface modeling program for the currently active SolidWorks Part. When you are finished surface modeling, choose **File>Exit & Return…** or just close the active window to return to the SolidWorks part. You may choose to save changes made in the Modeler and pass them back to SolidWorks, or not.

Remove SurfaceWorks Parent

SurfaceWorks>Remove SurfaceWorks Parent

[Only available when the selected entity has already been assigned as a SurfaceWorks parent.]

Removes (unassigns) SolidWorks geometry from being used as parent geometry for SurfaceWorks entities. You have the option to keep dependent SurfaceWorks geometry as "orphan" entities, meaning that they no longer have associativity with SolidWorks geometry; or you can opt to have the dependent SurfaceWorks geometry deleted from the model.

To remove SolidWorks geometry from use as SurfaceWorks parents:

1 In the graphic view, in the FeatureManager (this latter applies only to sketches, planes, reference curves, and user-named edges or planar faces), or in the Manage SurfaceWorks Parents dialog:

- Use the **left mouse button** to select one or more applicable entities, then choose **SurfaceWorks>Remove SurfaceWorks Parent** from the menus

  or
- Use the **right mouse button** to select the applicable entity and bring up the shortcut menu, then choose **Remove SurfaceWorks Parent**.

2 If a SolidWorks parent has SurfaceWorks dependents, you will be warned that removing the feature will orphan the SurfaceWorks dependents and you will have two options. Choose one by clicking its radio button:

Keep Dependents — the dependent SurfaceWorks geometry will be retained in the model but it will have no connection to its former SolidWorks parents, so any associativity will be gone

Delete Dependents — the dependent SurfaceWorks geometry will be deleted from the model

3 Click <**OK**> to continue with SolidWorks parent removal <u>and</u> SurfaceWorks dependent orphaning or deletion; click <**Cancel**> to abort and return to SolidWorks with parent assignments unchanged.

Note: When you remove edges or faces, the numbering of subsequent edges or faces listed in the Manage SurfaceWorks Parents dialog will change. In SurfaceWorks, a name is "attached" to a particular entity (you can edit it, but it won't change otherwise); in SolidWorks, edge and face enumeration is not "attached" to a particular edge or face.

SurfaceWorks Help Topics

Help>SurfaceWorks Help Topics

Opens a window that displays the Contents, Index and Search tabs for SurfaceWorks' online Help.

To access SurfaceWorks Help:

Choose **Help>SurfaceWorks Help Topics**.

Flattener

Flattener Introduction

Surface(s)

SurfaceWorks' Flattener, "Plex", is a compound-curved surface flattening utility based on AeroHydro's unique technology that was developed in 1988-89 and patented (U.S. and Canada) in 1990. The unique feature of the AeroHydro flattening technology is that it takes explicit account of the variable distribution of in-plane strain required to produce the specified 3D curvatures.

This solution enables accurate layout and NC cutting of flat blanks which, through application of a suitable compounding process, evolve into curved pieces precisely fitting their prescribed positions on the product.

The basic process is to design the 3D surfaces in SurfaceWorks, then use Flattener to calculate the flat shapes. Flattener output is in the form of DXF files that include (for each pattern):

- The flattened pattern outline
- Specified point and curve markings on the pattern
- A contour map of strain distribution
- The triangular mesh utilized for the solution

Surface(s)TM

In cases where there is a problem flattening a surface or a surface has been imported as a triangular mesh we offer an alternative flattening technology we call Triangular Mesh Flattening or "TMFlat". This employs different technology from Plex, but gives substantially similar results. Given a parametric surface, it first creates a Triangular Mesh and then flattens the result. It flattens a Triangular Mesh and a Composite Surface directly. The differences in the input data for these two methods are discussed below. The output differs in that additional strain information is given.

Plex vs. TMFlat

In this document, information can be considered valid for both methods unless indicated otherwise.

Compound Surface Forming and Compound Surface Flattening

Developable and Compound-curved Surfaces

Curved surfaces can be broadly classified into two categories:

Developable surfaces — those which can be rolled out flat (developed) onto a plane with no in-plane strain (shrinking or stretching) of any element. For example, all cylinders and cones are developable surfaces.

Compound-curved surfaces — all surfaces that are not developable. For example, any finite portion of a sphere is compound-curved — it cannot be flattened onto a plane without in-plane strain.

Gaussian curvature is a quantitative measure of the degree of compound curvature at a location on a surface. Gaussian curvature has units of $1 / \text{length}^2$. A developable surface is characterized by zero Gaussian curvature at all locations (Fig. 1, top). A compound-curved surface has non-zero Gaussian curvature at at least some of its points (Fig. 1, bottom). For example, a sphere has constant positive Gaussian curvature of $1 / R^2$, where R is the radius of the sphere. A saddle-shaped surface has negative Gaussian curvature (Fig. 1, bottom right). For further details about Gaussian curvature and surface curvatures in general, see "Surface Curvatures" and "Definition: Gaussian Curvature" in the "Editing Models" chapter.

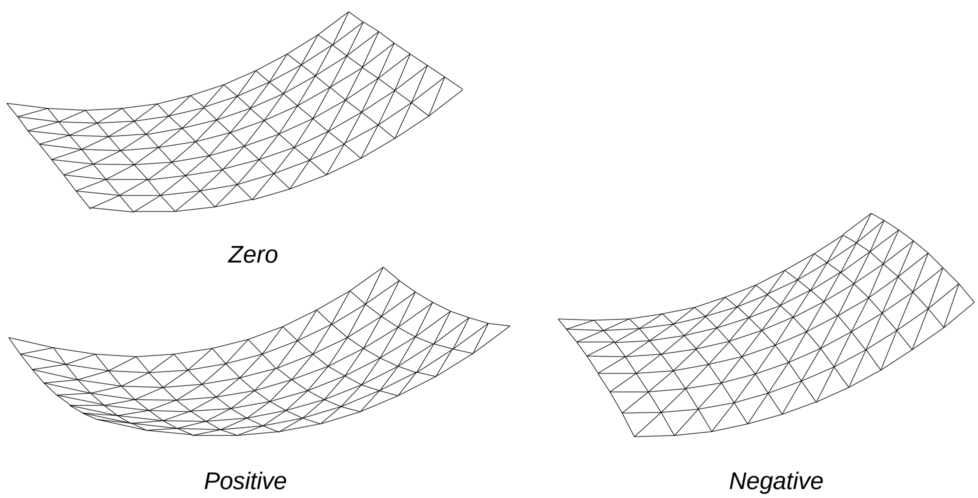


Fig. 1. Gaussian curvature: developable surface (top) and compound-curved surfaces (bottom).

You can use **View>Display>Surface Curvature** or  to discover the sign, magnitude, and distribution of Gaussian curvature of the surfaces you want to flatten.

Strain

"Strain" is used here in the sense of solid mechanics and electricity, to describe the fractional extension or compression of material going from the curved 3D surface to the flat plane of development. If the distance between two points is d in 3D and D in the 2D plane of development, the **flattening strain** is $s = (D - d) / d = D / d - 1$, i.e. the change in distance as a fraction of the original 3D distance. Positive strain means the distance increases during flattening. (Negative flattening strain is typical for surfaces having positive Gaussian curvature.)

Forming is the converse of flattening, so the strain required for forming has the opposite sign. We can define the **forming strain** as $s' = (d - D) / D = d / D - 1 = -s / (1 + s)$. When strain is small (compared with 1), s' is very nearly –s.

In general, strain can have different magnitudes (even different signs) in different directions. This is the case with anisotropic strain. Isotropic strain, in which a small circle drawn anywhere on the 3D surface turns into a small circle, generally of different size, on the flattened pattern, is a good description of some forming processes (such as peening and spherical die pressing), but it is clearly wrong for others (e.g. bias stretch). See "Flattening Parameters: Effects of Forming Process" on page 582.

In general, the larger the magnitude of Gaussian curvature (either positive or negative) in relation to the size of the work piece, the larger the strain that will be required for forming.

Forming Processes

Developable surfaces are relatively easy to manufacture, because they can be formed from flat sheet material by bending alone, with no in-plane strain required.

Compound-curved surfaces can only be formed from flat stock by processes that cause in-plane strain. A variety of forming processes can be used to induce in-plane strain, including:

Bias Stretching: Woven fabrics stretch diagonally along the bias.

Thermal Forming. Heating glass or metal and letting it stretch under its own weight or air pressure while cooling.

Assembly Under Stress: Elastic strain is induced by pulling a flat blank down onto a curved supporting frame.

Matched Die Pressing: The material is squeezed between matching male and female dies to form a complete curved piece in one stroke.

Spherical Die Pressing: Incremental pressing between generic dies having appropriate Gaussian curvature.

Explosive Forming: The material is forced into a female die by the pressure of an explosion.

Peening: Hammering on the material causes it to thin and expand.

Roller planishing: The material is passed between two roller wheels that are forced together, causing expansion and thinning.

Kraftformer: A press that grips thin material and can either expand or compress it in-plane.

Flattening

Flattening is the process of making a flat pattern (usually digital) from a 3D model. The pattern is used to cut a blank from flat stock and the flat blank is then subjected to a compounding process in order to form a curved piece.

The primary result of flattening is an outline for each individual flat blank. It is extremely valuable in the manufacturing process to have accurate flattened shapes, because accurate cutting is relatively simple while the material is flat, and having formed 3D parts which fit to each other accurately without trial and error or field trim greatly accelerates the assembly process. It is also valuable to be able to mark on the flattened blank the mapped positions of 3D features such as points (SurfaceWorks magnets) and curves (SurfaceWorks snakes).

Note: Marking a blank with magnets and snakes only works in Plex, as magnets, snakes and contours do not pass through in TMFlat.

Flattening Parameters: Effects of Forming Process

Because the mechanical transformation of the blank into a compound-curved plate involves in-plane strain, the plate expansion process needs information about how the strain is actually induced. (Any compound plate expansion program that doesn't require this information as input can't be doing an accurate job!) Two characteristics of the forming process are crucial:

Isotropy / anisotropy (Eccentricity parameter) **Plex only**
Expansion vs. contraction **Plex and TMFlat**

Isotropy / anisotropy (Eccentricity parameter)

First, let's think about how to characterize in-plane strain. From the perspective of the boat builder, strain is the material deformation in going from the flat blank to the 3-D curved surface of the boat. Imagine a small circle drawn on the blank — what does this turn into on the plate? The forming expands or contracts the circle, possibly by equal amounts in all directions, but also possibly by differing amounts in different directions. It all depends on how the forming is done. In general, the small circle becomes a small ellipse.

The special case where the ellipse has equal axes (i.e., it is a circle) is called isotropic strain (isotropic = "equal in all directions"; anisotropic is the opposite). Some forming processes have a basically isotropic character; for example, peening and spherical die pressing. Others are strongly anisotropic; for example, line heating produces practically all of its strain in the direction perpendicular to the heat line.

The case of roller planishing is of special interest because it is one of the most common methods of plate forming in small shipyards. In experiments we performed in conjunction with aluminum boatbuilder Lynn Davidson in 1990, we found roller planishing was strongly anisotropic, with a ratio of about 0.4:1 for the strain along the rolling direction compared with the strain across the rolling direction. That is, the wheel primarily pushes material to the side rather than along its direction of travel. This anisotropy can be allowed for in the plate expansion, or (depending on the size of the plate and the machine) it can be compensated for by rolling the plate approximately equal amounts in the longitudinal and transverse directions — this will produce approximately isotropic strain.

We characterize anisotropy by the Eccentricity parameter, the ratio of least to greatest strain. The Eccentricity parameter is 1 for an isotropic process, approximately 0.4 for unidirectional roller planishing, and 0 for line heating.

When the strain is anisotropic, the orientation of the forming process becomes important to the plate expansion calculation. We use the u-direction on the surface as a reference direction for indicating the orientation of the strain directions. One input to the plate expansion process is the strain angle, theta, from the u-direction to the direction of maximum strain.

Note: Strain angle has no effect unless the Eccentricity parameter is set to something other than 1.

Example: Roller planishing. Suppose the u-direction of the plate is the transverse direction on the hull and the plate is to be formed by rolling in the longitudinal direction. Then the direction of maximum strain is transverse, the same as the u-direction, so strain angle = 0 and Eccentricity parameter is 0.4.

Example: Line heating. Suppose the u-direction of the plate is the transverse direction on the hull and the plate is to be formed by transverse heat lines. Then the direction of maximum strain is longitudinal, so strain angle = 90 degrees and Eccentricity parameter is 0.

Note: **Sign convention for strain**. As noted above, from the fabricator's perspective strain is the deformation in going from the 2D blank to the 3D plate. From our perspective as the plate expands, strain is the deformation going from the 3D plate to the 2D blank. Therefore, in Flattener, positive strain means a contraction in the forming process, and negative strain means an expansion.

Expansion vs. contraction

The second critical characteristic of the forming process is whether it expands or contracts the material. Processes which can only expand are peening, spherical die pressing, and roller planishing. On the other hand, line heating can only contract the material. Kraftforming can do both, depending on the particular press head selected.

This possible constraint on the forming process is set on the Flattener Options tab in the Forming Process section. Choices are: Expansion only, Expansion + Contraction and Contraction only.

Reference

The most complete description of the underlying mathematics and finite-element solution used in the Flattener has been given in the following reference:

Letcher, J.S., Jr. : "Lofting and Fabrication of Compound-Curved Plates," Journal of Ship Research, Vol. 37, No. 2, pp. 166-175 (1993).

Designing Surfaces and Triangular Meshes in SurfaceWorks

As you create Flattener input surfaces and Triangular Meshes in SurfaceWorks, there are several issues to consider:

- Eligibility of a SurfaceWorks surface or Triangular Mesh to be flattened by Flattener includes shape and curvature characteristics. Some shapes just cannot be flattened, or have so much strain when flattened as to be not useful.
- For parametric surfaces, the number of divisions and subdivisions specified in SurfaceWorks affects the Flattener solution.

Plex

- The direction of the surface's positive normal in SurfaceWorks determines whether you'll specify positive or negative thickness in Flattener.

TMFlat

- When using Triangular Meshes you must orient them all the same way, (all normals pointing inwards, for example) in your model, and then tell the flattener which side you want to face up in the resultant flattened shape. TMFlat assumes that all of your normals are pointing in the same direction, and then writes the flattened solution to the DXF file, either all up or all down, as you choose. If your meshes have normals that face in opposite directions, your meshes will not appear in the DXF oriented in the same relative direction.

Eligible Surfaces and Triangular Meshes

Entity Characteristics

Your work for Flattener begins with a SurfaceWorks model. Flattener flattens one or more individual SurfaceWorks entities. In many cases, you'll be able to send your model's surfaces or Triangular Meshes to the Flattener without adjustment. But the Flattener does have certain requirements for eligibility:

- A surface or Triangular Mesh that is to be flattened should be smooth and continuous. If you need to have a crease (chine, knuckle), make the part with two smooth, continuous surfaces or meshes, one on each side of the crease.

- Additionally, a surface or mesh that will be flattened should not contain too high a degree of curvature. Regions of high curvature may have to be subdivided into smaller sections. When an entity must be broken up into two or more pattern pieces, use snakes to define seams and SubSurfaces between the snakes to represent the individual pattern pieces.

- An entity to be flattened should be a simply connected region, i.e. it must not contain holes. There are two general ways of dealing with a hole:

  1. If you are using Plex you can mark the boundary of a hole with a snake but not actually cut the hole until after compounding is completed.

  2. Using either Plex or TMFlat you can add a cut that connects the hole to the outer boundary of the surface or mesh. (The two sides of the cut become additional parts of the surface boundary.)

Developable Surfaces

Either flattening method will flatten a surface that is developable, i.e. has zero Gaussian curvature (the strain should come out essentially zero). Using MSDEV will give you more control of the result. To obtain MSDEV, contact AeroHydro.

Parametric Surfaces

Either flattening method may be used for flattening parametric surfaces. There is more control in the result using Plex, but if Plex will not work, a good result may be had using TMFlat.

Triangular Meshes and Composite Surfaces

TMFlat must be used for all Triangular Meshes and Composite Surfaces.

Surface Divisions and Subdivisions

The selection of divisions and subdivisions is important for accuracy and performance in the Flattener solution.

Plex uses a curvilinear triangular mesh division of each surface (each triangle side follows the curvature of the surface) for making calculations. Consequently, the accuracy of the solution depends on the degree of subdivision, in general improving rapidly with finer subdivision. At the same time, the computational cost (time) of the solution goes up rapidly, in proportion to the cube of the number of triangles.

TMFlat uses a linear triangular mesh division of each surface (each triangle side is straight) for making calculations. Because this is an inherently less accurate method of representing a surface, and computation time increases much less quickly with number of triangles for this method, we recommend increasing the number of triangles by a large amount if using TMFlat.

Triangular Meshes are only flattenable by TMFlat, and the number of triangles is not currently editable.

Composite Surfaces are only flattenable by TMFlat, and the number of triangles is editable.

Note: If a parametric surface and a Triangular Mesh or Composite Surface are both in the selection set, only TMFlat will be available.

Trimmed Surfaces

For information on flattening trimmed surfaces, see "Entity Descriptions - Trimmed Surface - Number of Triangles and Type of Triangulation.

Surfaces Other than Trimmed Surfaces

Divisions

On surfaces other than Trimmed Surfaces, the number of triangles used for flattening depends on the u- and v-divisions of the surface (note that u- and v-subdivisions are disregarded for this part of the process). The surface is divided first into a mesh of (curvilinear) quadrilateral panels along the visible division lines; then each quadrilateral panel is divided along one diagonal into two triangular panels. Therefore, the total number of triangles is 2 x u-divisions x v-divisions.

Example: Suppose a surface has 6x4 10x5 divisions – the number of triangles in the Flattener data will be 2 x 6 x 10 = 120.

Subdivisions

Subdivisions also have an important effect on the accuracy of the mesh data. This is because the triangle mesh is curvilinear, i.e. the triangles have curved edges. The surface needs enough subdivisions so Flattener can pick up the curvatures along the triangle edges and inside the triangles.

Recommendations Plex

Our recommendation is to use between 200 and 400 triangles per surface. Therefore, the product of u-divisions x v-divisions should be between 100 and 200. In addition, we would use at least 4 subdivisions in both the u- and v-directions.

Note: The surface flattening solution is a complex calculation; its time increases in proportion to the cube of the number of triangles.

Recommendations TMFlat

The number of triangles TMFlat creates from a parametric surface is calculated in the same way as Plex, but because there is much less penalty for more triangles and the triangles are made with linear sides, our recommendation is to use 5,000 or more triangles per surface. The quickest way to achieve this is to go to **Tools>Options>Performance>Model>Divisions Multiplier** and increase this an appropriate amount. In addition, we would use at least 4 subdivisions in both the u- and v-directions.

Example: If **Divisions Multiplier** is currently set at one, and there are 8 u-divisions and 4 v-divisions, set the **Divisions Multiplier** to at least 78 (5,000 / (8 x 4 x 2)).

Coordinating Divisions

For accuracy purposes, it is always valuable to coordinate the divisions between surfaces and their supporting curves. (See "Understanding SurfaceWorks Entities - Defining Entities - Additional Surface Properties - Coordinating Divisions"). If you are adjusting divisions and subdivisions for purposes of controlling Flattener data, be mindful of the divisions of any supporting curves or surfaces and change these also, if necessary, to keep them coordinated.

Direction of Thickness

Plex

Plex uses the direction of the positive normal of the surface to determine the direction of thickness on a surface. This direction has an impact on surface flattening only if a thickness is specified on the flattener's Surfaces tab.

The SurfaceWorks surface is one side of the blank. The blank's positive thickness direction is the same direction as the positive normal vector of the surface in SurfaceWorks, so check the direction of a surface's normal before running Flattener with thickness.

To check the direction of the positive normal:

In SurfaceWorks, select the surface.

- If you are running with Auto Orientation on (**Tools>Options>General >Auto Orientation**, and then either **Single selected entity** or **All selected entities**), the arrow indicating the positive normal will be present as soon as you select the surface.

- If you are running with Auto Orientation off, use **View>Entity Orientation** to display the normal arrow.

The positive normal appears as an arrow emanating from the center of the surface. Rotate the model if you cannot see it clearly.

Specifying thickness in Flattener:

- If the positive normal is pointing in the direction of thickness, enter thickness in Flattener as a positive value.
- If the positive normal is pointing away from the direction of thickness, enter thickness in Flattener as a negative value.

TMFlat

TMFlat does not handle thickness, however there is a way to simulate it if you are using it to flatten a parametric surface. Plex uses the thickness and direction of the normal to create an offset surface that is one half of the thickness in the normal direction from the base surface. This is the surface that is actually flattened. In this way, the strains are evened out (the neutral surface gets flattened). To simulate this for a parametric surface using TMFlat, create this same offset surface in your model, and send it through TMFlat.

Magnets and Snakes

Plex

Flattener will carry magnets, snakes and contours through the surface flattening process and mark them in their resulting positions on the flat pattern. The magnets, snakes and contours must be visible and must reside explicitly on the SurfaceWorks surface.

TMFlat

Magnets, Snakes and Contours do pass through to the flattened surface in TMFlat. If there is a need to show Magnets and Snakes on a Composite Surface, make a Surface Triangle Mesh from the Composite Surface and use TMMagnets and TMSnakes to be sure they transfer.

Using Flattener

The Basic Steps

Here are the basic steps you need to follow to use Flattener:

1. Design the input surfaces.
2. Set divisions, subdivisions, and Divisions Multiplier to get an appropriate triangle layout.
3. Select any combination of parametric surfaces and Triangular Meshes for flattening.
4. Start the Flattener by choosing **Tools>Flatten>Surface(s)** or **Tools>Flatten>Surface(s)TM**. If there is at least one Triangular Mesh in the selection set, only TMFlat will be available.
5. For Plex, fill out both tabs. For TMFlat, fill out the dialog box.
6. Click **OK**.
7. Name the file for output, click **Save**.

All surfaces treated the same for each run

There is no limit on the number of surfaces you may pick for flattening (other than your computer's memory), but note that all the surfaces in each flattening run are treated the same way. So for example, if you want one of your surfaces to have a different eccentricity parameter than the rest of your surfaces, it should be run separately.

Running Plex

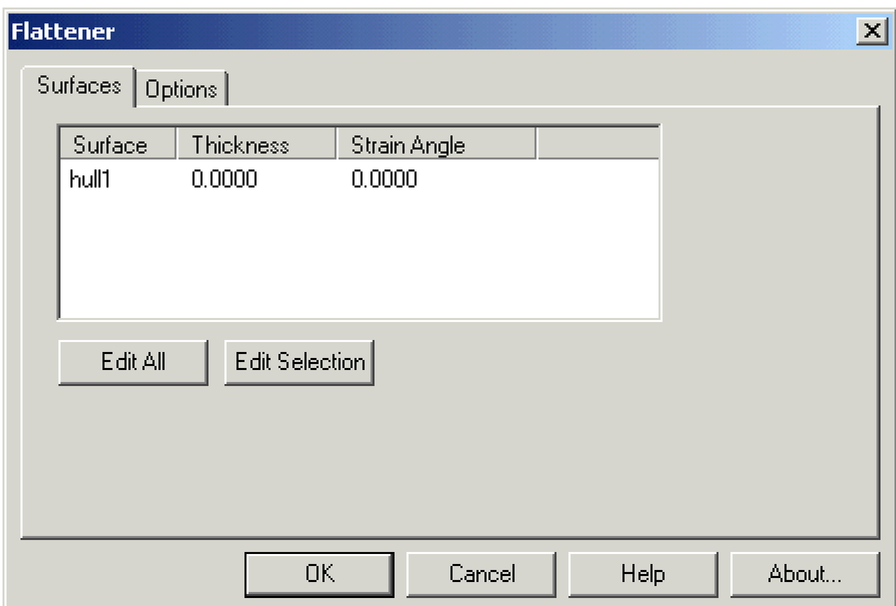
With a set of surfaces selected, choose **Tools>Flatten>Surface(s)**.

The Plex dialog opens with two tabs, "Surfaces" and "Options". Complete the data entries on the two tabs, then click OK.

OK runs the Flattener calculations. Cancel aborts the Flattener operation and closes the dialog window. Help is your access to Flattener Help. About gives version information about Flattener.

The Surfaces Tab

The Surfaces tab displays the names of the surfaces loaded from SurfaceWorks, along with each surface's Thickness and Strain Angle (both initially zero). Use the Surfaces tab to set the thickness and strain angle for each surface.



To edit thickness or strain angle:

1 Click **Edit All** or select one or more surfaces and click **Edit Selection** or double click one of the surfaces.

2 Enter the thickness if desired and / or the strain angle.

3 Click OK.

Selecting one or more surfaces for thickness and strain angle editing:

- To select a single surface, double click its name in the list box.
- To select several surfaces in an unbroken sequence, click the first one, then Shift+click the last one.
- To select surfaces in a disjoint sequence, click the first one, then Ctrl+click the others.

Thickness

"Thickness" specifies the thickness of each surface.

Input Values

Enter thicknesses as zero (no thickness) or a positive or negative decimal value, in the same units as used in the SurfaceWorks model from which the surfaces originated. The set of surfaces listed can have any combination of thicknesses (or thickness can be the same for all). When you first load a set of surfaces, the default assigned for thickness is zero.

When you edit thickness for a single surface, the default offered in the dialog is the thickness shown in the surfaces list box. When you edit thickness for a group of surfaces, the default offered is the thickness of the first surface in the selected set.

Discussion

When you specify non-zero thickness, Flattener uses direction of the surface's positive normal (from SurfaceWorks) to determine the direction of thickness. For details, see "Direction of Thickness" on page 586.

Strain Angle

"Strain Angle" specifies the strain angle of each surface.

Input Values

Enter strain angles as zero or a positive or negative decimal value between -90 and 90 degrees. The set of surfaces listed can have any combination of strain angles (or strain angle can be the same for all). When you first load a set of surfaces, the default assigned for strain angle is zero.

When you edit strain angle for a single surface, the default offered in the dialog is the strain angle shown in the surfaces list box. When you edit strain angle for a group of surfaces, the default offered is the strain angle of the first surface in the selected set.

Discussion

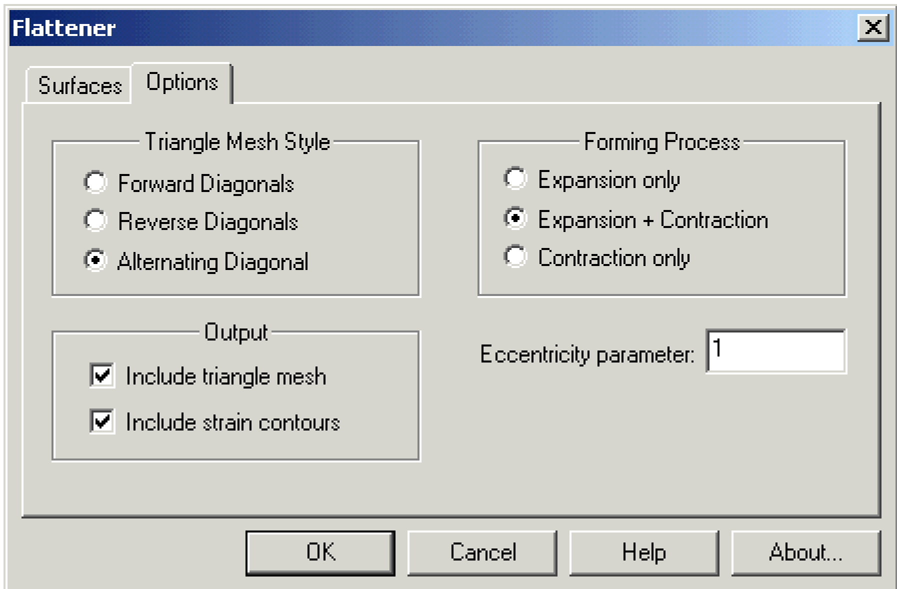
Strain angle is the orientation of the maximum strain direction with respect to the u-direction; default = 0. For details, see "Isotropy / anisotropy" on page 582.

The Options Tab

Use the Options tab to set:

- Triangle mesh style for the Flattener solution
- Forming process expansion / contraction constraint

- Output style
- Eccentricity parameter



Triangle Mesh Style

This option specifies the triangle mesh layout used for creating the input data.

Note: Triangle Mesh Style has no effect on Trimmed Surfaces.

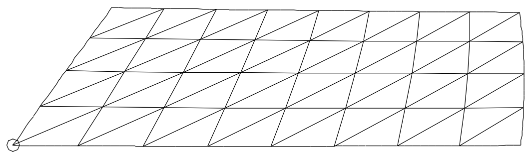
Options

Click the appropriate option button:

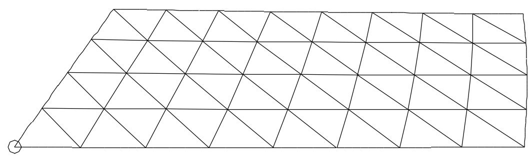
- Forward diagonals
- Reverse diagonals
- Alternating diagonals (the initial program default)

Discussion

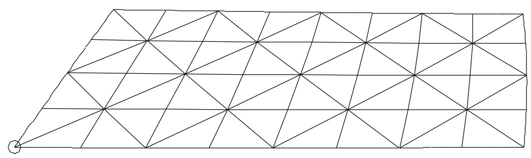
Flattener uses a curvilinear triangular mesh subdivision of each surface for making calculations. The triangles are generated by dividing the u-v quadrilateral mesh in SurfaceWorks with forward oriented diagonals, backward (reverse) oriented diagonals, or alternating forward and backward oriented diagonals (Fig. 2).



Forward diagonals



Reverse diagonals



Alternating diagonals

Fig. 2

The initial program default "Alternating diagonals" is the recommended first choice. If perchance you get a Solve error, try picking a different triangle mesh style and running the Flattener calculations again.

For details about how the triangle mesh is generated, see "Surface Divisions and Subdivisions" on page 585.

Forming Process

This option specifies the forming-process expansion / contraction constraint.

Options

Click the appropriate option button:

- Expansion only (e.g. for peening or spherical die pressing)
- Expansion and Contraction (e.g. for bias stretch or assembly under stress; this option is the initial program default)
- Contraction only

Discussion

A critical characteristic of the forming process is whether it expands or contracts the material. For details, see "Expansion vs. contraction" on page 583.

With the "Expansion and Contraction" option, arc lengths of edges will be accurately preserved during flattening, i.e. strain will be zero all around the boundary. This may or may not be true with the other forming process options, depending on the curvature of the surface.

Output

This option specifies the output you would like to generate in your DXF file.

Options

Click the appropriate option button:

- Include triangle mesh (default)
- Include strain contours (default)

Discussion

Default output for Flattener is an outline of the flattened shape of the surface, the triangle mesh used to generate the outline and the strain contours that result from the process. You can, if you wish, remove either the triangle mesh, strain contours or both.

Eccentricity parameter

This option specifies the Eccentricity parameter.

Options

The Options page opens in the Eccentricity parameter field. Make the number anything between 0 and 1 inclusive. Default is 1.

Discussion

For a discussion of the Eccentricity parameter see "Flattening Parameters: Effects of Forming Process" on page 582.

Other Buttons

These buttons are at the bottom of the Flattener dialog:

OK

Runs the Flattener calculations and creates the output files you specified. You will be prompted for the DXF file name.

The default DXF filename is the same as the filename of the SurfaceWorks file from which the surfaces were loaded, and the default location for the DXF file is in the same folder as the SurfaceWorks file.

As part of the solution process, Flattener creates a .PLX data file. The output filename and directory are the same as the DLX filename. This file is only created to facilitate our ability to troubleshoot problems you cannot solve. In the event that this situation arises and you contact us, we may ask you for information from this file. (For information on troubleshooting, see "Troubleshooting" on page 597).

Cancel

Aborts the Flattener operation and closes the Flattener window.

Help

Opens SurfaceWorks Help at either "The Surfaces Tab" or "The Options Tab" depending on the active tab.

About

Provides information about Flattener's version and build number.

Running TMFlat

With any combination of parametric surfaces, Triangular Meshes and Composite Surfaces selected, choose **Tools>Flatten>Surface(s)TM**. When the TMFlat dialog opens, complete the data entries, then click OK.

OK runs the TMFlat calculations. Cancel aborts the TMFlat operation and closes the dialog window.

Discussion

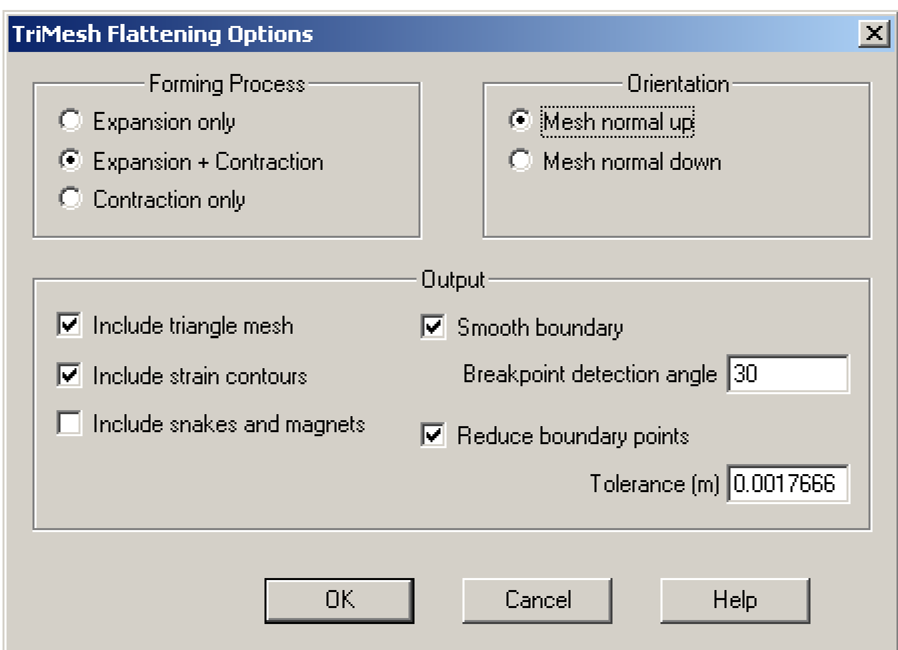
It is probably best to run TMFlat with **Smooth boundary** and **Reduce boundary points** off and see how the resultant flattened surface looks. If the edge looks jagged or otherwise not right, try using **Smooth boundary**. If the edge now looks better, check on the number of boundary points, (especially in any straight or nearly straight sections) and use **Reduce boundary points** if there seem to be too many. If that results in too few points, reduce the tolerance until you get a satisfactory result.

If all of that seems rather vague, it is because flattening needs are very material dependent. What works for steel may not be any good for cloth.

The TMFlat dialog

The TMFlat dialog sets the following:

- Forming process expansion / contraction constraint
- Orientation
- Output



Forming Process

This option specifies the forming-process expansion / contraction constraint.

Options

Click the appropriate option button:

- Expansion only (e.g. for peening or spherical die pressing)
- Expansion and Contraction (e.g. for bias stretch or assembly under stress; this option is the initial program default)
- Contraction only

Discussion

A critical characteristic of the forming process is whether it expands or contracts the material. For details, see "Expansion vs. contraction" on page 16-583.

With the "Expansion and Contraction" option, arc lengths of edges will be accurately preserved during flattening, i.e. strain will be zero all around the boundary. This may or may not be true with the other forming process options, depending on the curvature of the surface.

Orientation

This option specifies orientation of the flattened output in your DXF file.

Options

Click the appropriate option button:

- Mesh normal up(default)
- Mesh normal down

Discussion

You must tell TMFlat which side of your flattened surface you want to be facing upward (out of the screen) in your DXF file. Mesh normal up, the default setting, tells TMFlat to display the flattened surface as if you are looking at the normal up side. For more details see "TMFlat" on page 584.

Output

This option specifies the output you would like to generate in your DXF file.

Options

Click the appropriate option button:

- Include triangle mesh (default)
- Include strain contours (default)
- Smooth boundary (default)
- Reduce boundary points (default)

Discussion

Include triangle mesh and Include strain contours

Default output for TMFlat is an outline of the flattened shape of the surface, the triangle mesh used to generate the outline and the strain contours that result from the process. You can, if you wish, remove either the triangle mesh, strain contours or both.

Smooth boundary

TMFlat outputs the outline of the flattened surface as a polyline. Depending on your geometry and the number of triangles you use, this can have sharp angles in it. To reduce or remove these sharp angles, try using **Smooth boundary**. **Breakpoint detection angle** sets how smooth to make the resulting polyline.

Setting the **Breakpoint detection angle** too large will result in a polyline that is not very smooth. Setting this too low will result in a polyline with too many segments. Depending on your geometry, you may have to use trial and error to get results that you can use.

The SurfaceWorks default is **Smooth boundary** on and **Breakpoint detection angle** set to 30.

Reduce boundary points

Depending on the number of triangles you choose and the geometry of your surface, you can run into difficulty with your polyline consisting of a large number of small line segments. In a highly curved section this may be appropriate, but in straighter sections, this just makes it difficult for a cutting machine to deal with. Apply **Reduce boundary points** to alleviate this problem. **Reduce boundary points** uses the **Tolerance** value to remove some points. The larger the **Tolerance**, the more points are removed, but the more smoothed the boundary shape becomes. You must find the right balance between smoothing and shape.

The SurfaceWorks default is **Reduce boundary points** on and the **Tolerance** value set as follows:

Model Units	Tolerance
No units	.001
Inches / Pounds	0.0625 (1/16 inch)
Feet / Pounds	0.020833 (1/4 inch)
Feet / Short Tons	0.020833 (1/4 inch)
Feet / Long Tons	0.020833 (1/4 inch)
Millimeters / Grams	.01 (10 micrometers)
Centimeters / Grams	.001 (10 micrometers)
Meters / Kilograms	.001 (1 millimeter)
Meters / Metric Tons	.001 (1 millimeter)

Other Buttons

These buttons are at the bottom of the Flattener dialog:

OK

Runs the TMFlat calculations and creates the output files you specified. You will be prompted for the DXF file name.

The default DXF filename is the same as the filename of the SurfaceWorks file from which the surfaces were loaded, and the default location for the DXF file is in the same folder as the SurfaceWorks file.

Cancel

Aborts the TMFlat operation and closes the TMFlat dialog window.

Help

Opens SurfaceWorks Help at "The TMFlat Dialog".

Flattener Output

The default setting for **Output** places the following items in the output DXF, on different layers for each flattened surface.

Note: If you import this to SurfaceWorks to take a look, all the geometry will end up on one layer.

Flattened pattern outline (Fig. 3). The pattern outline is shown in bright magenta, on a layer named SurfName_Boundary, where SurfName is the name of the surface from SurfaceWorks.

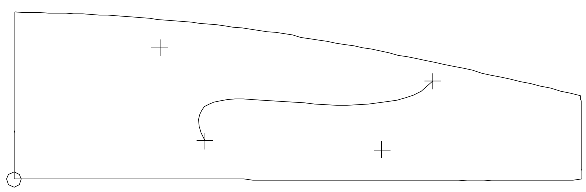


Fig. 3. Flattened pattern outline.

Images of magnets and snakes on the pattern for Plex. Magnets and snakes are shown in the same color assigned to them in SurfaceWorks. A magnet appears as a cross with the center corresponding to the magnet's position on the surface. They are located on a layer named SurfName_SnakesMagnets.

Contour map of strain distribution for (Fig. 4). The contour map of strain distribution is shown in four colors: bright blue, bright green, bright cyan, and bright red. Each color represents a different value of strain, which is labeled in the DXF file.

Positive strain means there is <u>contraction</u> of the blank during the forming process.

Negative strain means there is <u>expansion</u> of the blank during the forming process.

The maximum strain reported should be evaluated in relationship to the material and forming process. For example, if the maximum strain is shown as -.30 and the material you intend to use can't stretch 30%, you haven't found a useful solution. In this case, you could try cutting the surface up smaller.

The contour map is located on a layer named SurfName_StrainContours.

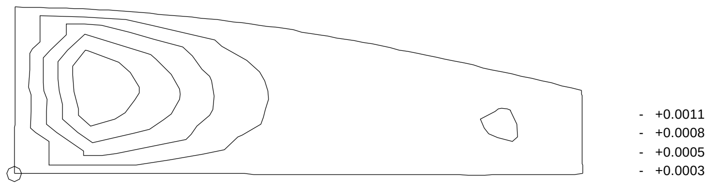


Fig. 4. Contour map of strain distribution for a surface.

Triangle mesh utilized for the solution (Fig. 5). The triangle mesh is shown in dark green and is located on a layer named SurfName_Triangles.

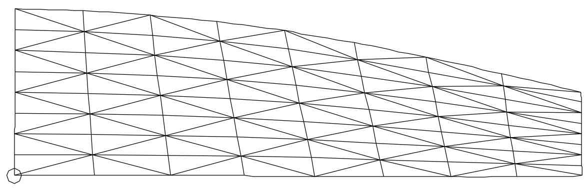


Fig. 5. Triangle mesh used for the Flattener solution.

Troubleshooting

Various error conditions can occur during the Flattener solution. If you get an error message, it usually will be accompanied by one or more fixes to try. The list below lays out the general steps to consider should you run into problems:

1 Review your surfaces — do any of them have particularly high curvature? If yes, use SurfaceWorks to create more, smaller surfaces in that area; then run Flattener again.

2 When you run Plex, use a different triangle mesh style — see "Triangle Mesh Style" on page 590.

3 In SurfaceWorks, change the surface divisions (or for a Trimmed Surface, change the "Number of Triangles" and/or change the divisions of the bounding snakes) — see "Surface Divisions and Subdivisions" on page 585. If your surface is already pretty finely divided, try fewer divisions; if your surface is only moderately divided, try more divisions. Run Flattener again.

4 Try making a Fitted Surface and flattening that instead.

5 Try making a Trimmed Surface and flattening that instead.

6 If you continue to get errors, contact us and we may be able to help.

What's New in SurfaceWorks 8.5

The new functionality added in the release of SurfaceWorks 8.5 is summarized as follows:

- 64 bit version available
- New Hydro program needed for 64 bit
- *UNIFORM Relabel
- Reprise License Manager
- New Command Window Commands
- IF Function bug fix

SurfaceWorks– Now in 64 Bit

To keep pace in the always changing software business we had to make the move to 64 bit architecture. This move was also needed to be compatible with SolidWorks 64. The SolidWorks integration has been revived with this move.

New Hydro Program

The Hydro program has been completely rebuilt. It may not look a great deal different, but much work has been done to enable it in the 64 bit environment. Some enhancements were added like better reports and printing, but the main goal was an up to date program, where enhancements can easily be added in the near future.

*UNIFORM Relabel

All curves and snakes have a relabel attribute, used to distribute the parameter t along the curve (or snake). The default relabel, '*', produces the "natural" labeling for the curve. This natural distribution of the parameter t along a particular kind of curve is described in the Entity Description for that curve (or snake).

In most instances, the default labeling is quite satisfactory. If you choose to relabel a curve, you must create a Relabel entity that specifies the new labeling and include the Relabel entity's name in the curve's definition.

Now there is another option. You can choose *UNIFORM, from the set of System Entities, as a relabel parent. The curve or surface will be relabeled with uniform t distribution along the entire arclength. This is a useful tool in controlling the U and V distribution on a surface. If the control curves are relabeled with *UNIFORM the UV lines of a surface will fall in a very predictable pattern. This is useful with lofted surfaces with multiple lofting curves or with curves with very different "natural" t-distribution. Relabeling parent snakes with *UNIFORM in a Blended Surface "fillet" is a good application as well.

Reprise License Manager

Built-in Licensing Software

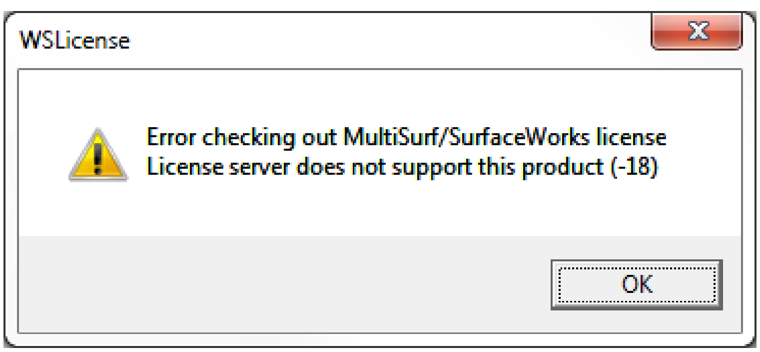
SurfaceWorks has a built in software licensing system to protect you and AeroHydro from unauthorized use of SurfaceWorks We have chosen this licensing system for its ease of use and the elimination of problems associated with hardware locking. Authorization is only an e-mail or phone call away.

Authorizing Your SurfaceWorks License

Your SurfaceWorks software comes preauthorized for a specified number of days (as found in your SurfaceWorks License dialog), the clock beginning when you first run SurfaceWorks After the expiration of the initial demo period, SurfaceWorks will not run until you have re-authorized your license. This means you can use the program immediately (or beginning at any time) for the specified period of consecutive days, but for continued use you must get your full authorization. You can apply for your full authorization at any time, even after the preauthorization period has expired. The basic process is: you give us your "Site Code," then we issue you an "Authorization Key" (details below).

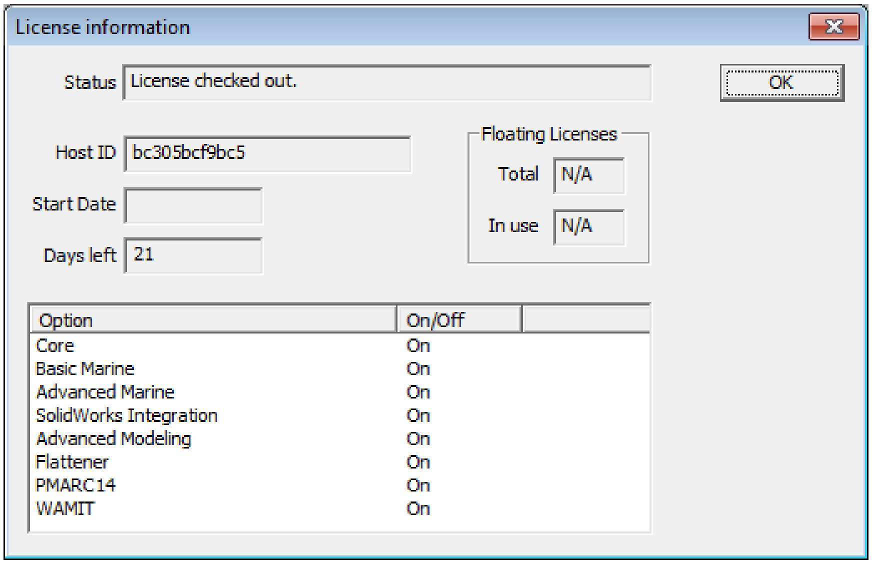
To Authorize Your License:

1a If your authorization or pre-authorization period has expired, the next time you start SurfaceWorks a message will appear on your screen:



Click <OK> to close the program. You may now continue with step **2** below.

1b If your pre-authorization period has not expired, choose **Help>Licensing**. The SurfaceWorks License dialog will appear.



2 **Look in the folder C:\Temp for the file 'hostid.txt'.** Email**, or fax, or your file to AeroHydro**. A license file will be returned with instructions of the proper location the file is to placed. (SurfaceWorks application folder)

Along with your hostid file, we also need:

- Program **name**, **Version #** and **Option Type** (i.e. SurfaceWorks , Advanced Marine, Flattener, ...). Program name and version can be found in **Help>About** SurfaceWorks or if this in not available, it can be found in the file **version.txt** found in the same folder as your program files.
- Your name

email: **support@aerohydro.com**

phone: **207-244-4100**

3 By authorizing your software you are agreeing to all terms of the software license agreement. You're now ready to roll with a fully authorized copy of SurfaceWorks !Welcome aboard.

Once authorized, you can review your current licensing status at any time by selecting **Help>Licensing**.

Note: Your SurfaceWorks license is site-specific to the particular folder on the particular machine in which you authorized it. If you want to move your copy to another machine, there is a specific procedure you need to follow; see "" on page 603. You may have your copy authorized in only one location at a time.

Network License Instructions

There are two install files. One for the Server and the other for the Clients. If the server needs to run MultiSurf/SurfaceWorks, both install files should be used.

Server:
32 bit: http://aerohydro.com/test/RLM_Server32.exe OR
64 bit: http://aerohydro.com/test/RLM_Server64.exe

1. Download the appropriate installation file for the OS on your server.
2. Run the install file which will create a sub-folder "AeroHydro\License Info" in your \Users\AppData folder and copy some files. If a different location is desired, you will prompted in the process for any change you require.
3. The install creates an RLM Server and installs it as a service. The service is started after the process.
4. Created in the previous step is a file called 'hostid.txt' in your C:\Temp folder.
5. Send the "hostid.txt" file to AeroHydro so it can be used to generate your permanent license file.
6. Go to this address on your web browser: http://<COMPUTER_NAME>:5054 to open the Server Admin Page.

7. Bookmark this address.

8. A license file will be returned with further instructions for the server and clients.

9. That should be it for the server.

Client:

1. Download and Install MultiSurf.

2. MultiSurf should run in evaluation mode for 21 days.

3. Included with a new license file for the server, will be an additional file for the client.

4. Please place this file in the MultiSurf / SurfaceWorks application folder.

5. MultiSurf / SurfaceWorks should open as a client.

SurfaceWorks is now enabled for software licenses that work over a network. In this scheme, a user group purchases a shared license for X number of users.

License Transferal

Transferring Your License: From retired computer

If you require your license to be moved from an old computer, destined to be retired for MultiSurf use, simply remove MultiSurf and all of its program folders. It will now be necessary to install MultiSurf on the new computer and apply for a replacement license file.

Transferring Your License: Temporary Transfer for Travel

Sometimes you want to transfer a SurfaceWorks license to a computer that already has a MultiSurf installed for the purposes of traveling. The way we facilitate this move is through an additional feature, called Roaming, of the Network License. An network license, along with its associated extra cost would be required.

The directions to allow roaming are as follows:

Let's call the computer which is to roam the "laptop". The process involves editing the computer's Environment Variables. (EV) These can be found here: Control Panel\System and Security\System\Advanced System Settings\Environment Variables

1. Close all instances of SurfaceWorks

2. With the laptop connected to the local network, either wired or wirelessly, open the EV dialog.

3. The RLM_ROAM EV is located with the "System" EV's.

4. Edit RLM_ROAM and change the value to the number of days you wish to roam.

5. Open SurfaceWorks on the laptop and check the license through Help / Licensing. You should see a reference to one roaming license checked out.

6. The laptop can be disconnected from the local network with no loss of the SurfaceWorks license.

7. To check the license back in:

8. Connect the laptop to the local network.

9. Change the RLM_ROAM EV to -1.

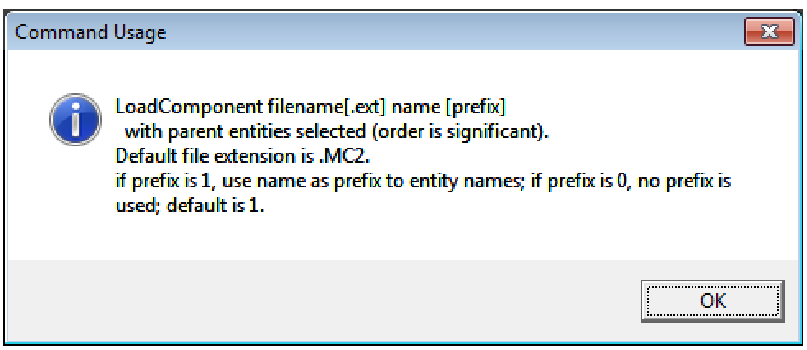
10. Start SurfaceWorks on laptop and an error will appear.

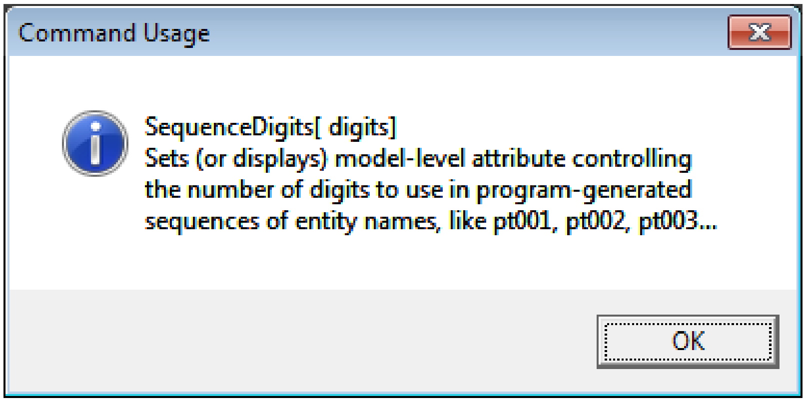
11. Close SurfaceWorks

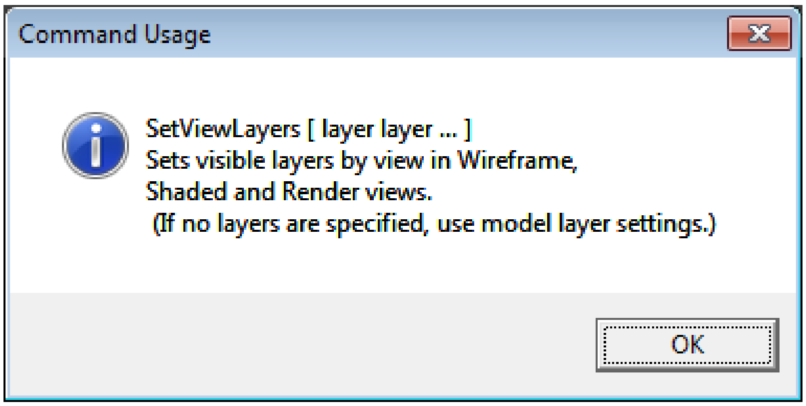
12. Change the RLM_ROAM EV to 0

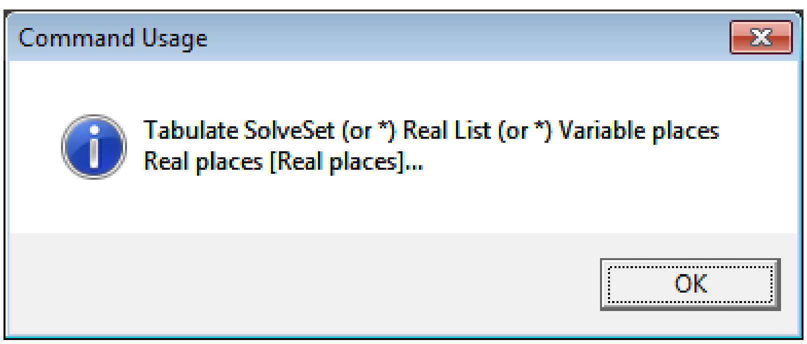
13. Open SurfaceWorks on the laptop and check license to see that normal network license has returned.

New Commands









IF Function Bug Fix

Any model that uses IF will need to be edited, reversing the order of arg2 and arg3, or the sign of arg1.

Glossary of Terms

ambiguous selection

When you click on a screen location to select an entity, and there are two or more entities within the cursor's "target" box that pass through the current filters, the selection is termed ambiguous. The Which Entity? dialog opens listing the eligible entities at that location, so you can indicate which one you want to select.

anisotropic strain

Anisotropic strain is strain that is not equal in all directions (the opposite of isotropic strain). Some forming processes are strongly anisotropic; for example, line heating produces practically all of its strain in the direction perpendicular to the heat line.

attribute

An entity attribute is any of the data items used to define the entity; e.g. name, color, visibility, divisions, type, relabel, orientation, supporting object(s), etc.

axial orthographic view

An orthographic view in which you are viewing along one of the world coordinate axes. The shortcuts for these views are: Front, Back, Left, Right, Top, Bottom (Mechanical View Orientation) and Bow, Stern, Port, Starboard, Deck, Keel (Marine View Orientation). When the grid is turned on, these views have both horizontal and vertical gridlines.

bead

A point constrained to lie on a curve.

body plan

(Nav. Arch.) One of the conventional naval architectural views of a ship hull: the two end half-views (orthographic) shown side by side, displaying the transverse sections of the ship.

boot stripe

(Nav. Arch.) A decorative and/or protective stripe near the resting waterplane of a boat, usually finished in a contrasting color to the hull surfaces above and below it.

bow

(Nav. Arch.) The front end of a boat.

B-spline

A continuous curve defined by a series of control points. The curve is formed in relation to the 3D polyline (i.e., the broken line) joining the control points in sequence. A B-spline always starts at the first control point and ends at the last control point, and it is always tangent to the polyline at these end points, but in general it does NOT pass through the other control points (as opposed to a C-spline, which passes through all its control points).

buttocks, buttock lines

(Nav. Arch.) A set of sections of a ship hull cut by vertical planes that are parallel to the centerplane; typically made with Contours parallel to the Y-plane.

camber

A measure of how much a curve deviates from its chord. Often used in different disciplines with specific meaning, e.g:

Maximum Camber - The greatest deviation of the curve from its chord.

Camber Ratio - The Maximum Camber divided by the chord length.

Deck Camber - Naval architecture term for Camber Ratio as measured for deck stations. It is frequently assumed that the shape of deck stations are arcs.

Foil Camber - The deviation of the Median curve from the chord, where the Median curve is the curve describing the points of mid thickness of the foil.

Camber Function - The camber as a function of chord length, usually for describing foils.

camera

You can think of the screen view of a model as being recorded by a camera, which is “where your eye is” in the 3D model space.

child entity

Any entity that is defined in terms of another “parent” or “support” entity. If you change any entity, the program automatically updates all of its children. Same as “dependent” entity.

chine

A “hard edge” between two surfaces; a join between two surfaces involving a sharp bend, hard corner, or break in slope. (Nav. Arch.) The join between two longitudinal hull surface strips, especially topside and bottom surfaces in a hard-chine boat hull; also, the structural member (if any) that runs along this join.

chord line

The straight line connecting the end points of a curve.

compound-curved surface

A compound-curved surface is any surface which must be stretched in order to flatten it onto a plane (i.e. any surface which is not developable). A compound-curved surface has non-zero Gaussian curvature at at least some of its points. Example: Any finite portion of a sphere is compound curved — it cannot be flattened onto a plane without in-plane strain.

contours

Contours are one or more parallel sections in a user specified orientation. Each section cuts all the surfaces desired for that contour creating 3D polylines that trace the surface(s) at that location.

control points

The point entities used to define the shape of a curve or snake entity.

control curves

The curve entities used to define the shape of a surface.

coordinate singularity

A location (u, v) on a parametric surface where the mesh of u,v coordinate lines fails to provide a unique normal direction, because of some form of collapse of the mesh.

- the rate of change of the surface point with respect to either u or v is zero, or
- the rates of change of the surface point with respect to u and v are parallel.

A pole is a common form of coordinate singularity. A place where the u- and v-directions are either identical or opposite is another kind of coordinate singularity.

Certain calculations break down at a coordinate singularity. The differential element of area associated with differential changes of u and v vanishes.

cove stripe

(Nav. Arch.) A decorative stripe that runs along the hull of a boat, not far below the sheerline, from near the bow to near the stern.

C-spline

A continuous curve defined by a series of control points. The C-spline curve passes through its control points in sequence (as opposed to a B-spline, which generally does NOT pass through its interior control points).

curve

A one-dimensional object, "straight" or not "straight". In SurfaceWorks, all curves (including Curves and Snakes) have points (including, as appropriate, Points, Beads, Rings and Magnets) as parents.

degenerate curve (or snake)

Normally a curve consists of a continuous 1-dimensional set of points. Under special circumstances, a curve can collapse into only a single point; for example, a line from point A to point A. Such a collapsed curve is called "degenerate".

degenerate edge

Normally each edge of a surface consists of a 1-dimensional continuous set of points. Under special circumstances, a surface edge can collapse to a single point; for example, a ruled surface between a curve and a point. Such a collapsed edge is called "degenerate".

degenerate surface

Normally a surface consists of a continuous 2-dimensional set of points. Under special circumstances, a surface can collapse into only a 1-dimensional set of points (for example, a B-spline Lofted Surface between two point entities), or just a single point (for example, a Tangent Boundary Surface made from four identically-located control points). Such a collapsed surface is called "degenerate".

dependent entity

Any entity that is defined in terms of another “support” or “parent” entity. If you change any entity, the program automatically updates all of its dependents. Same as “child” entity.

developable surface

Any surface which can be rolled out flat (or conversely, can be formed from flat sheet material) by bending alone, without in-plane stretching of any element. A developable surface has zero Gaussian curvature at all its points. Cones and cylinders are well-known examples.

digraph, directed graph

A diagram that represents the entities in a model (the nodes) and the dependencies between them (directed edges).

duck

(Drafting) A special weight used for positioning and holding a flexible curve or spline.

durable relationships

Durable relationships in a SurfaceWorks model are those which are built into the model through entity dependencies. For instance, if two surfaces depend on a common edge curve, the join between them is a durable join — you can change the shape of either of the surfaces, but they will continue to join along the shared control curve.

DXF file

An ASCII disk file which serves as a data exchange format between CAD programs. The DXF file standard is defined by Autodesk, but is used by essentially all CAD programs.

entity

A kind of geometric entity that a CAD system recognizes (for example, SurfaceWorks' Point, B-spline Curve, or Tangent Boundary Surface), or a specific entity made by specifying properties and / or parents for a kind of entity.

entity description

A reference topic in the manual or help system that lists the characteristic properties and / or parents required to create a particular entity, describes the role of those data items, and presents one or more examples.

Entity List

The Entity List serves as a container for a list of entities. This would be useful if, for instance, you wanted to perform repeated operations on a group of entities and wanted to be able to select this group repeatedly.

entity name

An entity name identifies a specific entity; e.g. 'top' might be the name of a Point located at X=0, Y=0, Z=10. Every SurfaceWorks entity has a name; entity names within a model must be unique.

exponential notation

Exponential notation, also called scientific notation, is a compact way to write very large and very small numbers. A number in exponential notation is written as one number, followed by an "e" or "E", followed by a second number. To read the number, you take the first number and multiply it by 10 raised to the power of the second number (positive or negative). Examples:

3.124E+009 = 3.124×10^{9} = 3.124 x 1,000,000,000 = 3,124,000,000

2.477E-005 = 2.447×10^{-5} = 2.447 x 0.00001 = 0.00002477

fair

adj. A fair curve or surface is one that is smoothly curved. It may have necessary inflections (e.g. a wine-glass cross-section), but it has no humps or hollows or abrupt changes in curvature.

verb To fair a curve or surface means to make it smoothly curved, removing any humps, hollows, or abrupt changes in curvature.

filter

A tool that allows only entities matching the filter to be available for selection (in graphics and Available Entities pane views). E.g. the Points Filter allows points to be available; the Current Layer Filter allows entities on the current layer to be available. Filters are additive.

Applying or changing filters does not affect entities already selected.

frame (entity)

A local orthogonal x,y,z coordinate system. A frame can be displaced and/or rotated with respect to the global coordinate system.

G0 continuity

Two curves or surfaces joining without tangency.

G1 continuity

Two curves or surfaces joining with tangency.

G2 continuity

Two curves or surfaces joining with tangency and continuous curvature across the junction.

generation (1st, 2nd, ... of parents, children)

When we talk about parents or children of an entity, we usually mean the “direct” or “first generation” of them. These are the parent entities you select in the Insert>Entitiy dialogs and the parent and child entities listed in the Parent/Child dialogs. But beginning with any one entity, parent (and child) relationships may continue on indefinitely, e.g: first generation parents, their parents (second generation), their parents (third generation), etc.

host entity

If entity A is embedded in entity B (another entity of the same or higher dimensionality), then entity B is the host of entity A. A bead has a host curve; a magnet has a host surface; a ring has both a host snake and a host surface. Converse of "resident".

IGES file

An ASCII disk file which serves as a data exchange format between CAD programs. The IGES file is an international standard which is supported by a large number of CAD programs. You can import IGES geometry with **File>Import>IGES**. Most of the surface and curve types and all supporting points are fully editable in SurfaceWorks. Although there is no Edit dialog for some entities that can be translated and imported into SurfaceWorks, these entities are relational entities with parents and you can change these entities in a more indirect manner.

inflection

A change in the direction of curvature; e.g. the point at which a curve changes from being concave to being convex.

internally-defined entity

An entity which is defined inside SurfaceWorks and which can be used as a support entity without a user needing to create it. Examples: ‘*’ (point at the origin), ‘*1’ (bead/ring at t=1 end of curve/snake), ‘*X=0’ (plane at X=0). Internally-defined entities don’t display in the drawing, so you must select them in the Available Entities pane. Same as “predefined” entity.

interpolation

1 Determination of (usually approximate) intermediate values between the entries in a table.

2 A curve that passes through a series of points is said to "interpolate" the points (for example, C-spline Curve). Likewise, a surface that passes through a set of curves or points is said to "interpolate" them (for example, C-spline Lofted Surface).

isotropic strain

Isotropic strain is strain that is equal in all directions (the opposite of anisotropic strain). Some forming processes have a basically isotropic character; for example, peening and spherical die pressing.

keel

(Nav. Arch.) An appendage on a sailboat hull which usually serves both ballasting and hydrodynamic functions. Typically it is a fin, approximately centrally positioned on the bottom of the hull, but it can be much more complex and specialized.

knot

Mathematicians generalize the definition of spline function to mean any piecewise polynomial function which satisfies certain conditions of continuity at the "knots", the points where the pieces join together.

latitude

One of the viewing angles used to control the camera position; angle above or below a horizontal plane.

lofted surface

A lofted surface is shaped by two or more control curves in much the same way as a curve is shaped by two or more control points. The process (which SurfaceWorks carries out automatically and invisibly) is:

1 Decide the orientation of the control curves with reference to the first control curve that has positive length.

2 On each of the control curves, locate points at the same parameter value t (or 1 - t, if the curve is used with reversed orientation).

3 Use the resulting series of points as the control points for a lofting curve.

lofting curves

In a lofted surface, the lofting curves are the curves “sprung” across the master curves to form the lofted surface; the u=constant parametric lines. The control points for each lofting curve are points at equal t-values (or 1 - t) on each of the control curves.

longitude

One of the viewing angles used to control the camera position and orientation; angle of rotation around the vertical axis.

longitudinals

(Nav. Arch.) The family of parametric lines that run fore-and-aft on a boat hull are sometimes called longitudinals. For example, these would be the u=constant parametric lines (the lofting curves) on a typical C-spline Lofted Surface hull with transverse master curves, like DEMO.MS2.

look point

The location at which the camera that records the view is pointed; the center of the orbit sphere when you rotate the camera. The default look point when you load a model is the center of the bounding box the program uses to frame the image on the screen (basically the center of the model). You can change the look point by panning.

magnet

A point constrained to lie on a surface.

master curves

The series of curves used to define the shape of a lofted surface.

mirror

The plane, line, or point used for reflection of a Mirrored Point, Mirrored Curve, Mirrored Surface; or for projection of a Projected Point or Projected Curve; or for specifying the infinite family of potential cutting surfaces for an Intersection Snake.

MS2 file

An ASCII disk file which defines a model created in SurfaceWorks.

MSDEV

AeroHydro's developable plate expansion utility. Will be provided upon request with SurfaceWorks Flattener.

normal, normal to

Perpendicular to. For example: a normal plane perpendicular to a curve; the normal direction perpendicular to a surface.

NURBS

NURBS is an acronym for "Non-Uniform Rational B-Spline." This is a formulation for parametric curves and surfaces which permits a uniform representation of B-spline, Bezier, and conic section curves and surfaces.

offset

A displacement perpendicular to a surface.

offsets

(Nav. Arch.) A conventional way of presenting a hull shape numerically, by giving transverse and vertical coordinates at a series of stations.

orbit sphere

When you rotate the camera, it works as though it were mounted on the surface of a sphere, pointing inward toward the center which is the look point. This imaginary sphere is called the orbit sphere of the camera.

orientation (of curve or surface)

The direction in which the parameters t (curve / snake) or u,v (surface) increase. Entities such as Developable Surfaces require the orientation of one or more basis curves / snakes to be in a specific direction. View>Entity Orientation displays this information for the selected entity.

orthographic view

An orthographic view maps a 3D scene into a 2D screen image using parallel projection lines. This is an artificial view which is conventional for most engineering drawings.

parameter

A dimensionless real number which labels one point along a curve; or one of a pair of real numbers which label one point on a surface. In SurfaceWorks, curve parameters are called t and surface parameters are called u and v; all nominally run from 0 to 1, but have meaning outside this range.

parametric lines

The curves on a surface that result from holding one of the surface parameters (u or v) constant and varying the other.

parent entity

An entity that another entity directly depends on for its shape or position. E.g. the curve on which a Relative Curve is based; one of the control curves of a lofted surface; the surface from which a SubSurface is built. Same as “support” entity.

perspective view

A perspective view maps a 3D scene into a 2D screen image using projection lines which converge on an eye or camera point. This is a natural view which is similar to a visual image or a photograph.

plan view

A vertical orthographic projection of a scene (from above or below). (Nav. Arch.) A vertical orthographic view of a hull, displaying the waterlines.

plane

An unbounded flat surface. In SurfaceWorks there are default planes (*X=0, *Y=0 and *Z=0) that consist of the points that define X=0, Y=0 and Z=0 respectively and planes with any orientation that you can create in various ways.

point

A non-dimensional location in space. In SurfaceWorks, points are 2 parent entities, each defined in relation to a Point and a Frame.

pole

A degenerate edge of a surface. At a pole, one set of parameter lines (u = constant or v = constant) converges to a point. For example, the apex of a cone; or either pole of a sphere made by revolving a semicircle about the line joining its endpoints. A pole is a common form of coordinate singularity.

polyline

A broken line consisting of 2 or more straight-line segments joining a series of points. Polylines can be 2D (the points have 2 coordinates, usually X and Y) or 3D (the points have 3 coordinates).

predefined entity

An entity which is defined internally in SurfaceWorks and which can be used as a support entity without a user needing to create it. Examples: '*' (point at the origin), '*1' (bead / ring at t=1 end of curve / snake), '*X=0' (plane at X=0). Predefined entities don't display in the drawing, so you must select them in the Available Entities pane. Same as "internally-defined" entity.

profile, profile line

(Nav. Arch.) The outline of a hull in side view, particularly the edge of the hull surface at the centerplane.

profile view

A horizontal orthographic view of a scene (from either side). (Nav. Arch.) An orthographic view perpendicular to the hull centerplane, displaying the profile and buttock lines.

property (of an entity)

An entity property is any of the non-parent data items used to define the entity; e.g. name, color, visibility, divisions, type, degree, etc.

Relabel (entity)

A Relabel entity controls the relabeling (re-parameterization) of a curve, snake or some surfaces. It is not a visible entity in itself, but its effects on labeling can be made visible by showing tickmarks for the affected curve or snake. Its effect on surfaces is harder to see, but magnets will change their absolute location on a surface as the surface's u-v parameterization changes.

Relational Geometry

Also referred to as RG. The conceptual framework of SurfaceWorks: a scheme for defining geometric entities and the relationships between them. Capturing relationships as part of a geometric model allows automatic propagation of changes to all affected entities, and supports the key advantages of Relational Geometry: accuracy, design flexibility, parametric design, ability to freely refine and optimize.

resident

If entity A is embedded in entity B (another entity of the same or higher dimensionality), then entity A is a resident of entity B. A bead resides on a curve; a magnet resides on a surface; a ring resides on both a snake and a surface. Converse of "host".

ring

A point constrained to lie on a snake. A ring also lies on the surface the snake is on, so it can serve as a magnet on that surface.

RMS (root-mean-squared)

A measure of the deviation between two sets of data.

scientific notation

See "exponential notation".

selection set

The selection set is comprised of the one or more entities currently selected. Its basic use is: you make a selection set, then you tell the program what you want to do to it (edit, move, hide, use as parents for an entity, ...).

sheerline

(Nav. Arch.) The upper edge of a hull surface; usually the most visually prominent line on a boat.

SLDCRV file

An ASCII text file containing X, Y, Z values in three columns. Each line represents one 3D point, and must contain three decimal values separated by spaces or tabs. Can be opened in SurfaceWorks with **File>Import>SLDCRV**.

snake

A curve constrained to lie on a surface.

spline, drafting spline

(Drafting) A thin, flexible curve held in position by weights ("ducks"), used for drafting free-form curves.

stations

(Nav. Arch.) A set of longitudinal locations on a ship, usually uniformly spaced; also, the transverse sections at these locations.

stem

(Nav. Arch.) The forward edge of a boat hull. (Also, a structural member along this edge.)

stern

The back end of a boat.

support (or supporting) entity

An entity that another entity directly depends on for its shape or position. E.g. the curve on which a Relative Curve is based; one of the control curves of a lofted surface; the surface from which a SubSurface is built. Same as “parent” entity.

surface

A 2-dimensional object, i.e. an object that can be defined by 2 parameters. In SurfaceWorks, all surfaces have curves (Curves or Snakes) as parents.

t (parameter)

t is the parameter used to reference/label curves and snakes. It runs from 0 at the starting end to 1 at the other end of the curve or snake. You can think of each point that forms the curve as being labeled with a unique value of t.

thwart

(Nav. Arch.) A seat extending from side to side of a small boat.

tilt

One of the viewing angles used to control the camera orientation; rotation of the camera about the line from the camera to the look point.

transom

The distinct surface (if any) that forms the back end of a boat.

u and v (parameters)

u and v are the parameters used to reference/map surfaces. From the 0,0 corner of the surface, u runs from 0 to 1 in one direction along the surface; v runs from 0 to 1 in the other direction. You can think of each point that forms the surface as being labeled with a pair of values of u,v.

waterlines

(Nav. Arch.) A set of sections through a hull, cut by a family of horizontal planes; typically made by one or more Contours parallel to the Z-plane.

WireFrame (entity)

A WireFrame entity is a .3DA or .PAT file which can be included in a model primarily for display or comparison purposes. The result of including this file is a wireframe entity which cannot be used in the construction of other SurfaceWorks entities. For example, a curve represented in a WireFrame entity cannot support a bead, and it cannot be used as an edge of a surface.

wireframe image

An image of a model in which surfaces are represented by meshes, curves and contours by polylines, and points by little dots. A wireframe is essentially transparent — you can see between the wires, and when part of a surface is behind another surface, you can still see it.

zero-velocity point

You can think of the parameter t on a curve as time and the curve as being the path of a moving point as t goes from 0 to 1. The parametric velocity of the point is the rate of change of its position with respect to t.

A zero-velocity point (t-position) on a parametric curve or snake is a place where the parametric velocity vanishes. At a zero-velocity point, a small change in the parameter t causes no change in the 3D location of the moving point. This can be caused by a variety of situations, including: doubled control points on a degree-2 or lower B-spline Curve, a point serving as a curve, an ellipse with a zero major or minor axis. Some calculations will fail at a zero-velocity point — for example, the tangent direction is not defined.

Appendix

Recovery Files

Description

Note: Recovery files are not available for SurfaceWorks in SolidWorks Integration mode.

Autosave files (.buX) protect you from losing data due to unexpected events (such as power failure) that cause SurfaceWorks to stop functioning without going through its normal close down procedure.

Recovery files (bu1 through bu4) on the other hand, can protect you from those (very) rare and awkward situations where SurfaceWorks has closed normally, but the data saved in the file is not what you really intended (e.g. you inadvertently changed decimal places to 1, rather than the divisions multiplier, then saved the file).

Each of these files takes the name of the model file and the appropriate extension. .bu1 is the only file that exists when the model is closed normally. The other three will exist if a model file has been opened and a fatal program crash has occurred (unlikely as that is!) sometime after the model has been successfully loaded. The files get written to the same directory as where the model file exists.

I want my model back

If there has been a crash of some sort, and you cannot open your model or it has been corrupted in some way:

Autosave files:

1 **Rename** your model file, **name.ms2, to name.old.**

2 **Rename name.bu(latest) to name.ms2.**

3 **Open name.ms2.**

4 If this file does not help either, the best you can do is open one of the recovery files.

If you realized you saved the model to a state you would rather not be in and you want to go back to a previously saved state:

Recovery files:

1 **Rename** your model file, **name.ms2, to name.old**

2 Rename name.bu(latest) to name.ms2.

3 Open name.ms2

Output File Specifications

Overview

This section details the output file specifications for: .3DA, .OFE, .OF4, .RUL.

.3DA — 3D ASCII Drawing File

A .3DA file contains wireframe and / or panel information in ASCII format. It consists of an unlimited number of records each in the format:

pen% x y z

pen% = integer controlling pen color; 0 = pen up
x, y, z = world coordinates of point

Spaces can be used for separators.

A panel is represented in a .3DA file by a group of 3 or more records having the following special characteristics:

pen% = 0 for the first point
pen% > 0 for other points
x, y, z of first and last point are identical

.OFE — Offset File (older format)

File is ASCII. Each record is terminated by CHR$(13) + CHR$(10)

First record: np, id$

np = no. of records (points + station identifiers) to follow
id$ = identifying message, no commas

Rest of file is np records, each with 2 decimal numbers, free format.

Second record: ns, idum

ns = no. of stations
idum = dummy, not used

Rest of file is ns stations. Each station consists of a “station identifier” record plus npts point records.

Station identifier record: x(i), npts(i)

x(i) = X-position of station x 100
npts(i) = no. of points in this station

Point record: z(i,j), y(i,j)

z(i,j) = vertical coordinate x 100
y(i,j) = horizontal offset x 100

No terminating record is required.

Stations have to be in order of increasing X-position.

Z is positive upward; Y positive to starboard.

Only one half of a station is represented. (Bilateral symmetry is assumed in an .OFE file).

First and last point in a station must have Y = 0.

Points go around half-station counterclockwise when viewed from astern: beginning at the base of the keel and continuing up the starboard side of the hull, across the deck, etc., and back to centerline.

.OF4 — Offset file (newer format)

An .OF4 file is a representation of one or more solids by a series of closed polylines giving their transverse (X = constant) sections. The sections can be arbitrarily spaced, and the points along each section are arbitrarily positioned and are independent of other sections. The .OF4 file is intended primarily for hydrostatic and hydrodynamic analysis.

File is ASCII. Each record is terminated by CHR$(13) + CHR$(10). Items in a record are separated by one or more spaces.

First record: a three character string, “OF4”.

Second record: id$

id$ = the identifying string for the file, up to 255 characters in length, no commas

Third record: 0 (zero; reserved for future use)

Fourth record: UnitsPerFoot, Unit$

UnitsPerFoot = a real number specifying the relationship between the Offset File’s units and feet (used to facilitate scaling, etc.)

Unit$ = a two character abbreviation for the name of the file’s units (“??” if unknown)

Fifth record: NumStas%

NumStas% = an integer specifying the number of stations in the Offset File, ranging from 1 to 32767

There follows NumStas% stations, each starting with the following record:

StatNum, LSym%, XPos, NumPts%

StatNum = a real number, the user-defined label for that station

LSym% = an integer specifying station symmetry, -1 = symmetric, 0 = asymmetric

XPos = a real number, the X-position of the station

NumPts % = a positive integer, the number of points in the station

Each station includes NumPts% points, each consisting of the following record:

Colr%, pLabel$, z, y

Colr% = an integer specifying the color of the line drawn to the point; colr% = 0 means pen up

pLabel$ = a two character point label, with " ~~ " specifying no label
z = the point's Z-coordinate, positive upward
y = the point's Y-coordinate, positive to starboard

- No terminating record is required.
- Stations have to be in order of increasing X-position.
- If a station is symmetric, only half the station is represented, otherwise the full station is stored.
- Points go around the station counterclockwise when viewed from the stern, starting at the base of the keel, and continuing up the side of the hull, across the deck, etc.

.RUL — Ruling File

A Ruling File is basically a table of ruling endpoint coordinates for one or more surfaces.

File mode is random; record length = 24.

All numbers are single precision, MBF format.

Each surface is represented by a set of records as follows:

Records 1 and 2: ID message for the surface; up to 48 characters, padded with spaces.

Remaining records consist of 6 fields, each 4 characters long (X1$, Y1$, Z1$, X2$, Y2$, Z2$).

Record 3:

X1$ = number of rulings (NR)
Y1$ = 0
Z1$ = 0
X2$ = 0
Y2$ = number of snake points (NS)
Z2$ = orientation (0.0 = normal; 1.0 = reversed)

Records 4 to 3+NR:

X1$, Y1$, Z1$ are the coordinates of the first end of a ruling
X2$, Y2$, Z2$ are the coordinates of the second end of the ruling

Records 3 + NR + 1 to 3 + NR + NS:

X1$ = pen / color:
0 if pen up
> 0 if pen down
if > 255, Z1$ + X2$ + Y2$ + Z2$ is an object name (Y1$ not used):
if 256, this is a Contours object
if 257, this is an XContours object
if 258, this is a YContours object
if 259, this is a ZContours object
if 260, this is a CvContours object
if 261, this is a snake object
if 262, this is a magnet object

if 263, this is a ring object
Y1$, Z1$ = u,v where point is located
ruling no. = 1 + u * (NR - 1)
fractional distance along ruling = v
X2$, Y2$, Z2$ = X,Y,Z coordinates in 3D space

Program Limits

SurfaceWorks stores its data in highly-flexible dynamic structures, so there are few hard limits on anything. But there are a few ...

Object name: maximum 16 characters.

Line length in model file: maximum 8192 characters+spaces (but we don't recommend coming even close to this limit!)

Total curve divisions:

divisions x subdivisions = 160,000.

Total surface divisions:

u-divisions x u-subdivisions x v-divisions x v-subdivisions = 160,000. (An easy-to-see example: 20x20 x 20x20.)

Divisions multiplier (Tools>Options, Performance, Model) = 1 - 10

In general, the divisions limits are FAR in excess of what is needed for adequate screen images, and even for actual building patterns. We feel the program default divisions (8x4 for curves and 8x4 8x4 for surfaces) are sufficient for viewing most curves and surfaces, at least initially.

Error Codes

Overview

SurfaceWorks handles two principal classes of errors during its operation:

1. Model File Errors: while reading a SurfaceWorks model file from disk.

2. Geometry Errors: while evaluating the model geometry.

These kinds of errors do not threaten the functioning of the program — you fix the error(s), then continue right on with your work. SurfaceWorks lets you know what is wrong by showing an error flag, , in the Entities Manager next to the entity in error. If you right click an entity in error and choose **What's Wrong**, a window opens displaying a diagnostic **Error Code** (number) and a brief notation of what the code means. In this appendix, we:

- explain how SurfaceWorks handles the two classes of errors

- list the error codes, in numerical order, along with some explanatory notes

Model File Errors 100 - 199

Model file errors are errors in the format of the disk file in which the model is saved. Many of these are trapped while the model file is being read in and transformed line-by-line to its internal representation. For example, the program knows when a line starts with an unrecognizable entity keyword or when an object name used in an entity definition has not been assigned to an object. In these cases, the program displays, in the graphics window, the objects in the file that were successfully read, and in the Error Manager, each entity in error with an error flag.

SurfaceWorks should not find model file errors in a file that was created and edited interactively within the program; it is supposed to be smart enough to prevent that. Such an error may be evidence of a program bug, therefore we request that you send us the problem file and a description of the steps taken to create it.

102 The entity type is not recognized by this program version

An entity description line begins with a word (an entity keyword) which is not the name of a valid SurfaceWorks entity. Check the spelling of the entity keyword. Entity keywords are not case sensitive. A given version of SurfaceWorks may not support all the entity types listed on the reference card or in the “Entity Descriptions”.

103 The entity name is already in use

The second item in an entity description is the entity name. Entity names have to be unique. Apparently you are trying to assign to a second entity a name you already used. Note, entity names are case sensitive.

104 Unexpected semicolon encountered; entity data was incomplete

The semicolon ; is used to end each entity in a model file. Apparently a semicolon was encountered before all the data for an entity was read in. Most likely you have left out one or more data items. This error can also occur if you omit the closing brace } in an entity which uses braces { }.

105 Expected semicolon not found; entity has excess data

The semicolon ; is used to end each entity in a model file. Apparently all the data for an entity was complete, but the next thing in the file was not a semicolon. Check the entity description. Most likely, you are supplying too much information or you have omitted the ; needed to terminate an entity.

106 At least one entry is required in these braces {...}

Braces { } are used in the model file to surround lists of variable length, e.g., the list of control points for a BCurve. There is no situation where the list can be empty. *EMPTY* is used as a placeholder in situations when the braces would otherwise be empty.

107 An expected open brace { was not found

Braces { } are used in the model file to surround lists of variable length, e.g., the list of control points for a BCurve. At the point where the program expects such a list to occur, something other than the opening brace was found. Check for extra numbers in the data (typically color, visibility, divisions, type, etc.) that precedes the {.

108 Invalid use of the default name '*'

'*' can serve in place of an entity name for reference to some entity classes, such as point, curve, frame, knotlist, or relabel. Defaults do not exist for all entity classes, however; for example, '*' is an invalid reference for a bead, a line, or a surface.

109 One or more of this entity's parents were left undefined (or misspelled)

Every entity used as parent for another entity must be defined somewhere in the model file. Check the spelling (including capitalization) of the parent names; entity names are case sensitive.

110 An entity is the wrong class of entity to serve as a parent

There are specific rules about which SurfaceWorks entities can serve in various roles in creating an entity (check the entity description). For example, a line can serve as a curve, but a curve can't serve as a line, for very logical reasons.

111 The specified kind index is invalid for this entity

Many entity types have a kind as part of their makeup — an index selecting spline order, a foil section, a method of construction, etc. Each entity has its own range of valid kinds, which are checked during read-in. Check the entity description for valid kind numbers.

113 ' * ' is not a valid name for this entity

' * ' can be used as the "default entity", serving as a point, curve, frame, graph, knotlist or relabel in various circumstances. It is not being used correctly in the present case.

114 Unexpected end of file

Apparently the file ended in the middle of a line.

115 The visibility index is invalid for this entity type

Valid ranges for visibility indices depend on the entity class: points, curves, surfaces, etc.

116 Invalid statement in header portion of file

A file line occurring before the BeginModel statement could not be understood.

117 File line length exceeded a program limit

See "Program Limits" on page 17-625 for limits on line lengths in files.

118 Value out of range (0-255)

Some integer values in the model file are limited to this range (one byte).

119 This entity was used as a parent, but was never defined or created

Create the entity so named, or specify a different support.

120 File has later language version than supported

The first line of a model file specifies the language version number. The file being read in has a later number than your program version, and may be incompatible. Consider upgrade to current version.

121 Slash character (/) is missing in this line

The slash is essential punctuation in the file line for every entity type.

EXPAND THIS

122 Entity name has one or more invalid characters

A limited number of characters are valid for use in entity names: letters, digits, some punctuation marks. See "Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Name"

123 Line ends before entity name

A file line ended unexpectedly, apparently with only one token, an entity type.

124 Unexpected open brace ({); probable missing data

Braces are used in the model file to surround lists of variable length. A { character was found in an invalid position.

125 Unexpected close brace (})

Braces are used in the model file to surround lists of variable length. A } character was found in an invalid position.

126 Unexpected semicolon (;); probable missing data

The semicolon ; is used to end each entity in a model file. Apparently a semicolon was encountered before all the data for an entity was read in.

127 Unexpected slash (/); probable missing data

The slash is essential punctuation in the file line for every entity type. It has to be in the right position, following optional data such as user data, layer number and attributes.

128 Unexpected open parenthesis (; probable missing data

There is no valid usage of parentheses in the model file except within model-level and geometry-level comments, entity user data, and attributes.

129 Line ends before color

For most entities, a color index is the third data item. Apparently the file line is incomplete.

130 Line ends before visibility

For most entities, a visibility index is the fourth data item. Apparently the file line is incomplete.

131 Required divisions x subdivisions are missing

Curve entities require one divisions x subdivisions specifier (e.g. 16x2), surface entities require two (e.g. 8x4 10x2); solid entities require three (e.g., 4x2 5x2 6x2);.

132 Required orientation index is missing

Surface entities, and some other entity types, require an orientation value (0 or 1).

133 Unexpected end of line; probable missing data

The semicolon ; is used to end each entity in a model file. Apparently a semicolon was encountered before all the data for an entity was read in.

134 A tagged data value uses an undefined tag

Tagged data items are in the form `tag:value`. Only a limited number of tags are defined. This error can be caused by entity user data without quotes.

135 The specified model has one or more cyclic dependencies

According to the logical relationships specified in the file, some entity depends indirectly on itself (and that is not allowed).

136 Divisions error: 0 is invalid for divisions or subdivisions

Division and subdivision values must both be positive integers (1 to 255).

137 The number of characters in the entity name exceeds a program limit

Entity names are limited to 16 characters.

138 Color index out of range (0 to 255)

See "Understanding SurfaceWorks Entities - Defining Entities - Properties Common to All Entities - Color."

139 Orientation different from 0 or 1

Surface entities, and some other entity types, require an orientation value before the `/`. 0 and 1 are the only valid values.

140 Divisions error: (divisions x subdivisions) could not be read

A divisions specifier is two integers (1 to 255) with the letter x between; no spaces (e.g. 8x4).

141 Divisions error: divisions out of range (1 to 255)

The first integer in a divisions specifier (before the x) is out of range (1 to 255).

142 Divisions error: subdivisions out of range (1 to 255)

The second integer in a divisions specifier (after the x) is out of range (1 to 255).

143 A supposed tagged item token lacks a colon (:)

Tagged data items are in the form `tag:value`. The colon is required. This error can be caused by entity user data without quotes.

144 Entity user data lacks a closing quote

Entity user data must be enclosed in double quote marks (“ ”). If the closing quote is missing, the program tries to read the rest of the line as part of the user data. This error will also occur if you try to embed double quotes within a user data comment.

145 Length of entity user data exceeds limit

The limit is 40 characters. See also "Program Limits" on page 17-625.

146 *0 or *1 used for parent other than bead or ring

System entities *0 and *1 can only be used to specify beads and rings.

147 u-divs x u-subd x v-divs x v-subd exceeds limit

The program limit is 160,000 for this product (an example: 20x20 x 20x20). See also "Program Limits" on page 17-625

148 Divisions multiplier out of range

The valid range is 1 - 10.

149 A system plane is not permitted in this position

The system planes *X=0, *Y=0, *Z=0 can only be used as plane parents.

150 Unsupported or unreadable model units

The string specifying model units could not be understood.

151 Unsupported or unreadable model symmetry specification

The string specifying model-level symmetry could not be understood.

152 Model home view latitude out of range (-90 to 90)

Use Tools/Options/ Model and specify home view with Lat between -90 and 90.

153 Model home view longitude out of range (-180 to 180)

Use Tools/Options/ Model and specify home view with Lon between -180 and 180.

154 Model decimal places specification less than 0; taking default (3)

If you want to change from the default 3 places, use Tools/Options/Performance/ Decimal places; range is 0 to 6.

155 Model decimal places specification above limit (6); taking default (3)

If you want to change from the default 3 places, use Tools/Options/Performance/ Decimal places; range is 0 to 6.

156 Model file layers on / off specification unreadable

Use Tools/ Layers to reset layers on/off, then save your model again.

157 Model divisions multiplier less than 1; taking default (1)

If you want to change from the default 1, use Tools / Options / Performance / Divs. multiplier; range is 1 - 10.

158 Model divisions multiplier over limit (10); taking default (1)

If you want to change from the default 1, use Tools / Options / Performance / Divs. multiplier; range is 1 - 10.

159 Fewer than 6 values specified for model extents

Use Tools / Options / Performance and click <Update Extents>, then save your model again.

160 The specified model extents are impossible; (high < low)

Use Tools / Options / Performance / and click <Update Extents>, then save your model again.

161 Expected a file name

This error happens when a file name is expected (such as a .3DA file for a TabCurve), but none is found where expected.

163 Expected one data type and got another

This happens when, say, a character string between double-quotes is expected and an integer is found instead, or a decimal number is expected and a character string is found. Check the entity to be sure that all its data is specified, and in the correct order.

164 Unknown application name in model file header

All model files must begin with an application name and language version number. Evidently at one time the application name was checked; it no longer is, so this error should not occur.

165 Cannot interpret hexadecimal number

WindSurf is expecting to read a hexadecimal number and is unable to recognize the token as such.

166 Cannot interpret integer

This error would occur, for instance, if you had entered 12a or 12.4 when the program was expecting an integer (e.g. 12).

167 Cannot interpret real number

This error would occur, for instance, if you had entered 12.1.0 or 12a when the program was expecting a real number (e.g. 12.1 or 12).

168 Character string expected but not found

The program is looking for (and has not found) a pair of double quotes surrounding a character string. This error would occur, for instance, if you had entered abc instead of “abc”, or “123 instead of “123”, or ‘point 19’ instead of “point 19”.

169 Cannot interpret entity name

When reading the model file an entity name was expected but another type of token was encountered.

170 Odd number of characters in a supposed hexadecimal string

Hexadecimal strings must have an even number of hexadecimal digits.

171 Hexadecimal sequence is too long

Break the hexadecimal sequence into two or more shorter even-length sequences of hexadecimal digits.

172 Invalid character in a hexadecimal sequence

Be sure your hexadecimal sequences (e.g. for layer on/off settings) include only the legal characters: 0 through 9, a through f, and A through F.

173 Wrong data specified for an attribute value.

174 Invalid attribute property.

175 An attribute is defined without a type.

176 An attribute is defined with more than one type.

177 Cannot write file to disk; possibly out of disk space

SurfaceWorks was unable to save the file to the disk (drive or floppy) specified. Make more disk space available, then try to save the file again.

178 Entity type not supported by this program version

Different versions of SurfaceWorks support different sets of entity types. The entity type in question is not supported by the program version you are running. This error is most likely to happen if you have received a model from another user who has a more recent or more capable version of the program.

179 The layer number is invalid

Valid layer numbers are 0 to 255

180 File name has more than 255 characters

File name must be no more than 255 characters.

181 Function in formula has too few arguments.

Each function requires a specific number of arguments. The function can't be evaluated when one or more arguments are missing.

182 Function in formula has too many arguments.

Each function requires a specific number of arguments. Excess arguments are treated as an error.

183 First argument of function has the wrong type.

The first argument of a function in a formula does not have the correct type. For example, when the argument is supposed to be an entity, you cannot use a real constant.

184 Second argument of function has the wrong type.

The second argument of a function in a formula does not have the correct type. For example, when the argument is supposed to be an entity, you cannot use a real constant.

185 Third argument of function has the wrong type.

The third argument of a function in a formula does not have the correct type. For example, when the argument is supposed to be an entity, you cannot use a real constant.

186 A formula is too long or complex.

During internal processing, a formula has exceeded limits on length or complexity. Try breaking the formula up into smaller formulas.

187 Units mismatch between operands of + .

The units of the operands of a sum (+) do not match. The two numbers added must have the same unit dimensions.

188 Unit power overflow or underflow.

A *, / or ^ operation would yield a unit power outside the range -127 to +127.

189 Units mismatch between operands of - .

The units of the operands of a difference (-) do not match. The two numbers subtracted must have the same unit dimensions.

190 Invalid fundamental unit.

Unit dimensions must be expressed as a combination of integer powers of L, M, T or I.

191 Invalid Variable range: Upper bound is below the lower bound.

For example, a Variable entity specified as : "Variable f2 / 4.5 L (6, 3) ;" has its bounds crossed.

192 Invalid format for variable range.

The proper format for a Variable entity with a range is: "Variable f2 / 4.5 L (3, 6) ;".

Either upper or lower bounds, or both, can be missing in the parentheses, but the comma is required.

193 The units of the formula's result differ from the units specified.

The units of the result of the evaluation of a formula differ from the units specified for this Formula entity. For example, this error will occur when the formula is the product of two lengths, and the Formula units are different from area (L^2).

194 Real parent has incorrect unit dimensions.

The entity has a real parent in its definition which is not of the right units. For instance:

"AbsPoint ... / 0 y 0 ;" requires that y have length units;

"AbsBead ... / curve t ;" requires that t be unitless.

195 Function arguments have mixed unit dimensions.

In a function (such as MIN) where all arguments must be real with the same units, the actual arguments were not all of the same units.

196 Missing operator inside formula.

For example, the following formula definition gets this error: "Formula f3 / { 2 3 } ;".

(There is no operator between 2 and 3.)

197 Missing open brace { in Formula syntax.

In model file syntax for a Formula entity, the expression must be enclosed in braces {...} .

198 Missing close brace } in Formula syntax.

In model file syntax for a Formula entity, the expression must be enclosed in braces {...} .

199 *UNIFORM used for parent other than a knotlist.

The system entity *UNIFORM can serve only as a knotlist. It is being used here in some other role.

200 Minus sign encountered in unexpected place.

Negation (unary minus) can only be applied to real constants and real parents.

201 *00, *01, *10, or *11 used for parent other than a graph or magnet.

These system entities can be used for a graph, and under some circumstances for a magnet, but they can serve only in those two roles.

Geometry Errors 201 - 250

Geometry errors are errors which occur during the evaluation of model geometry. Geometry errors come in several flavors. Some are places where a division by zero would occur — the program checks before dividing, and graciously issues an error message instead of crashing. Some result from logical defects in the model — for example, a snake using control magnets that are on different surfaces. Some come about when iterative calculations would go on forever without finding a solution — for example, a complex intersection that fails to converge.

When a geometry error is encountered, SurfaceWorks displays the entities that evaluate correctly in the graphics display, and the entities in error are shown in the Entities Manager with an error flag, and in the Error Manager with an error code and an explanatory message.

Generally, the best fix is to edit the entity in error to correct the geometry. You may need to select and correct a parent entity that is causing the error. Refer to "Entity Descriptions" to see what is needed for your entity to conform to its entity specification.

Note that many of the errors may be number 284 "Entity is in error because one or more of its parents is in error". Fixing the parent entity(s) listed that have errors other than 284 error will usually allow the child to evaluate.

201 Program math error; should not occur. Please report this to AeroHydro.

This catchall error should occur only as the result of incorrect program logic. It is not your fault. Please report such an error to us in detail for program correction (see "Getting Started - Technical Support").

202 Arc kinds 1,2,3: coincident points; arc is indeterminate

If any pair of supporting points is identical (and the other point is different), SurfaceWorks cannot determine what plane the arc is supposed to lie in.

203 Arc kind 1: points lie on a line, not in 1-2-3 sequence.

Arc type "Arc through 3 points". If this type of Arc is made with 3 points on a line, the points must be picked in order.

204 Arc kinds 2,3,6: points lie on a line; arc is indeterminate

The type "Arc: start-center-end guide" and type "Circle: start-center-plane guide" arcs use point3 to establish the plane the arc (circle) should lie in. If the three points all lie on a line, SurfaceWorks cannot determine a plane.

205 Surface has too many divisions

There is a limit to the total number of sub-divisions for any surface. In this case that means the (u-divisions x u-sub-divisions x v-divisions x v-sub-divisions). That number has been exceeded. You must either reduce one of these values or break up the surface into smaller surfaces.

206 Zero divisions for a curve or snake

Any curve or snake requires at least one division for tabulation and display.

Both values must be at least 1.

207 Zero divisions x subdivisions in u-direction of surface

Both values must be at least 1.

208 Zero divisions x subdivisions in v-direction of surface

Both values must be at least 1.

209 Improper sequence of control points for an X-spline.

Control points must be in either ascending or descending order with respect to the "Direction" property. For example, if the Direction property is "X orientation", the control points must be in order of either ascending or descending X coordinate.

210 C-spline: Coincident control points

A C-spline passes through its data points. If two successive points are identical, the calculation fails. This can happen on a CCurve when the same point is named twice or when two successive points happen to come out with identical coordinates. It can happen on a CSnake when two successive magnets have identical u,v positions. It is also possible on a CLoftSurf, if master curves cross.

211 C-spline: Singular matrix in solution

This error should never occur; it is not your fault if it does. Please report the occurrence to us (see “Getting Started - Technical Support”).

212 A foil must have 3, 4, or 5 control points

Foils can be constructed with 3, 4, or 5 control points — 3 for a half section, 4 for a symmetric full section, or 5 for an asymmetrical or symmetrical full section. SurfaceWorks does not accept any other number of control points for a Foil Curve, Foil Snake, or Foil Lofted Surface.

213 CLoftSurf: Crossed master curves, between u=0 and u=1

For at least one u-value being evaluated on the real portion of the surface, the lofting C-spline has coincident control points. Revise the master curves so they don’t cross.

214 CLoftSurf: Crossed master curves, for u<0 or u>1

For at least one u-value being evaluated on the extension of the surface, the lofting C-spline has coincident control points. Revise the master curves so they don’t cross.

215 PolyCurve labeling error; transition t values must be strictly increasing.

This error pertains to a second variety of PolyCurve (PolyCurve entity type), where part of the data is an explicit specification of the t parameter value where each junction between component curves occurs -- the so-called "transition t values". These values must be in an ascending sequence.

216 RadiusArc: Coincident control points.

The various kinds of radius Arc can't get a solution when certain combinations of their control points are coincident.

217 Plane2: Coincident points; plane is indeterminate

A 2 Point Plane requires two distinct points. The line from point1 to point2 defines the normal direction to the plane. If point1 and point2 are identical, SurfaceWorks cannot determine a plane.

218 Plane3: 3 points are on a line; plane is indeterminate

A 3 Point Plane requires three points. If the three points lie on a line, SurfaceWorks cannot determine a plane. One way this can happen is for any two of the points to be identical.

219 RadiusArc, type 3: Zero angle at point 2.

Kind-3 Radius Arc with points 1 and 3 in the same direction, and positive radius.

220 Projection axis line has zero length; direction is indeterminate

For projection onto a line mirror, the line has to have nonzero length; otherwise, it does not define a direction.

221 Rotation axis line has zero length; direction is indeterminate

For rotation about an axis line, the line has to have nonzero length; otherwise, it does not define a direction.

222 NURBCurve, NURBSnake: Too few knots in specified knotlist.

A NURBS curve or snake has too few knots for its degree and number of control points. If N is the number of control points and D is the polynomial degree (e.g., 3 for cubic), then a KnotList needs exactly N-D+1 entries, the first being 0 and the last being 1. A KnotList2 needs exactly N+D+1 entries, the (D+1)-th being 0 and the (D+N-1)-th being 1. Example: if N=5 and D=2, a KnotList needs 4 entries such as { 0 1/3 2/3 1 }; a KnotList needs 8 entries such as { 0 0 0 1/3 2/3 1 1 1 }.

223 NURBCurve, NURBSnake: Too many knots in specified knotlist.

A NURBS curve or snake with too many knots for its degree and number of control points. See error 222.

224 NURBSurf: Too many zero weights; zero divisor

Depending on its type, a NURBS Surface may tolerate one or more zero weights, but too many in the same area can cause a divide by zero. Zero weights are not recommended in any case.

Another cause of this error is extremely closely spaced knots, sometimes seen in imported NURBS geometry. There is a command "FixNURBKnots" that can usually fix this problem.

See “Understanding SurfaceWorks Entities - Collective Entity Information - NUB and NURBS Entities.”

225 NURBSurf: Invalid ncu (number of control points in u-direction)

ncu (number of control points in the u-direction) tells a NURBS Surface how many rows there are in its rectangular mesh of control points. This can never be less than 1, or more than nc (number of control points).

226 NURBSurf: nc not evenly divisible by ncu

nc (number of control points) for a NURBS Surface is found by counting the points in braces. ncu (number of control points in the u-direction) is read from the data before the braces. To make a rectangular mesh, nc must be evenly divisible by ncu.

227 Conic: Primary axis has zero length; curve is indeterminate.

The primary axis of a conic has zero length (points 1 and 2 coincide).

228 IntPoint has duplicate parents; no solution.

Intersection Point entity has duplicate parents; this cannot produce a distinct intersection.

229 IntPoint: Solution failed, equations are singular.

Singular equations for Intersection Point. Make sure the three surfaces or planes do intersect. You can use a starting magnet close to the solution, in place of a surface parent.

230 Fillet: Parent snakes 1 and 2 are on different surfaces.

Blend Surface: Blend boundary 1 and Blend helper 1 are on different surfaces.

231 Fillet: Parent snakes 3 and 4 are on different surfaces.

Blend Surface: Blend boundary 2 and Blend helper 2 are on different surfaces.

232 IntPoint: failed to converge.

An Intersection Point generally requires an iterative solution of three simultaneous nonlinear equations. The process failed to converge in the allowed number of steps. This often indicates a nonexistent or poorly specified intersection. Try using a starting magnet near the intersection in place of any surface parent.

233 Helix: axis line has zero length; direction is indeterminate.

The axis line for a helix must have non-zero length, in order to establish a direction.

234 NURB Curve, Snake or Surface: Too many identical knots.

A NURBS curve of polynomial degree D (e.g., cubic is D = 3) can have up to D identical interior knots, but any more than that leads to a division by zero in the evaluation of the B-spline basis functions. Example: quadratic B-splines (D=2) with N=6 control points; {0 .25 .50 .75 1 } and {0 .25 .50 .50 1} are valid KnotLists, but {0 .5 .5 .5 1} is not -- it has more than D identical knots.

235 NURBCurve: too many zero weights; zero divisor

Depending on its degree, a NURBS Curve or NURBS Snake may tolerate one or more zero weights, but too many in the same area can cause a divide by zero. The error occurs when the number of sequential zero weights is greater than or equal to degree. Zero weights are not recommended in any case.

Another cause of this error is extremely closely spaced knots, sometimes seen in imported NURBS geometry. There is a command "FixNURBKnots" that can usually fix this problem.

See “Understanding SurfaceWorks Entities - Collective Entity Information - NUB and NURBS Entities.”

236 KnotList: Knot sequence must be non-decreasing.

Each value in a KnotList must be greater than or equal to the preceding value.

237 Index for list selection must be positive.

Zero or negative index for list selection entity. The index must be 1 to N, where N is the size of the Entity List parent.

238 RelSurf: Wrong number of points (1 or 4 required).

A Relative Surface must have either one or four point parents.

239 OffsetSurf: Wrong number of offsets (1 or 4 required)

The valid number of corner offsets for an OffsetSurf is either one or four. If only one offset is given, the offset is constant.

240 SubCurve: Specified beads are on different curves

The two beads delineating a SubCurve must be on the same curve.

241 SubSnake: Specified rings are on different snakes

The two rings delineating a SubSnake must be on the same snake.

242 SubSurf: Specified snakes are on different surfaces

The two or four snakes that outline a SubSurface must all be on the same host surface.

243 Index for list selection points beyond the end of the list.

Index out of range for list selection entity. The index must be 1 to N, where N is the size of the Entity List parent.

244 SubSurf: Wrong number of snakes (2 or 4 required)

The valid number of snakes outlining a SubSurface is either two or four. If the number of bounding snakes is 2, the resulting SubSurface is the portion of the basis surface between the two snakes (effectively a “RuledSubSurf”). If the number of bounding snakes is 4, the resulting SubSurface is the region bounded by the four snakes (effectively a “BlendSubSurf”).

(A Trimmed Surface can have any number of bounding snakes.)

245 BlendSurf: Wrong number of boundary curves (4 required)

The only valid number of boundary curves for a blended surface is four. To make a BlendSurf with three sides, you can use a point for one boundary curve.

246 DevSurf: Minimum of 16 for u-divisions x u-subdivisions

For a Developable Surface, the product of u-divisions x u-subdivisions must be at least 16.

247 XLoftSurf: Crossed master curves, between u=0 and u=1.

248 XLoftSurf: Crossed master curves, for u<0 or u>1.

249 Blister: Snakes on different surfaces.

Snake1 and Snake2 must be on the same surface.

250 Control magnets for snake are on different surfaces

All the magnets defining a snake must be on the same surface.

Geometry Errors 251 - 300

251 RelSnake: magnet1 on different surface from basis snake.

For a Relative Snake, both magnet parents must be on the same host surface as the Snake parent.

252 RelSnake: magnet2 on different surface from basis snake.

For a Relative Snake, both magnet parents must be on the same host surface as the Snake parent.

253 PolySnake: component snakes are on different surfaces

The different component snakes making a PolySnake must be on the same host surface. (Under appropriate circumstances you can make a PolyCurve from snakes on different surfaces.)

254 PolySnake labeling error; transition t values must be strictly increasing.

This error pertains to a second variety of PolySnake (PolySnake entity type), where part of the data is an explicit specification of the t parameter value where each junction between component snakes occurs -- the so-called "transition t values". These values must be in an ascending sequence.

255 The entity selected from an EntityList is in the wrong class for its required service.

An entity selected from an Entity List by a List Select entity is in the wrong class or subclass; for example, a curve has been selected in a situation where a point is needed.

256 SweepSurf using default guide: path is too highly curved.

Computing a default guide requires sufficient continuity in the path curve (continuous slope, or G1).

257 SweepSurf: Evaluating at a zero-velocity point on path.

Computing a default guide can fail when the path curve has zero velocity at some parametric locations.

258 SweepSurf: Guide point lies on tangent to path.

The moving frame for a SweepSurf cannot be evaluated at a u-position where the point on the guide lies along the tangent to the path.

259 SweepSurf: Frame is indeterminate.

The moving frame for a SweepSurf could not be oriented from the data computed from the specified path and guide.

260 IntSnake: No intersection found

SurfaceWorks failed to find any intersection between the cutting entity and the entity being cut. Check your selection of supporting entities, and try to visualize the cutting surface for this case. Try to reposition the magnet parent so it is closer to where you expect the intersection to start.

261 IntBead, IntRing, IntRing2: Singular equations.

The problem posed for an intersection had no definite solution. This can occur, for example, when the curve for a kind-1 XYZBead is parallel to the plane X=0. Review the kind and other data to be sure you have specified the intersection you want. Use a bead near the desired intersection for a better starting value.

262 IntBead, IntRing, IntRing2, XYZBead, XYZRing: Failed to converge.

Solution for an intersection is an iterative process. In this case the iteration failed to converge to a solution within required accuracy. Review the kind and other data to be sure you have specified the intersection you want. Use a bead near the desired intersection for a better starting value.

263 ProjSnake: Projection failure

For at least one point on the parent curve, the projection line has zero length or indeterminate direction. This can happen, for example, when the mirror is a line, and the curve being projected intersects the line at some point.

264 ProjSnake: Coordinate singularity on surface

The search for a point of intersection started out, or ended up, at a place on the surface where a normal could not be calculated. This is usually at a degenerate edge or corner of the surface, e.g. a three-cornered surface.

265 ProjSnake: Singular equations

The search for an intersection point failed. This can happen when the intersection does not exist (e.g., intersection of a line with a parallel plane); or when the search starts at a coordinate singularity on the surface, or a point where the surface is parallel to the line of projection or intersection. Try moving the magnet parent closer to where the Projected Snake is supposed to start.

266 IntSnake or ProjSnake: Failed to converge

The search for an intersection point failed to reach a firm solution in the allowable number of steps. This can happen when the intersection does not exist (e.g., a projection line which approaches the cutting surface but does not actually intersect it); when the projection line intersects the surface at a very low angle; or when the starting point for the search (the seed magnet) is too far from the true intersection point. Review the entities used to specify the intersection and try to visualize the projection lines. Try creating / moving the seed magnet.

267 ProjSnake: mirror surface different from basis surface

When mirror/surface is a surface, it has to be the same surface that the projected snake lies on.

268 PolySurf: u values must be strictly increasing.

Part of the data for a PolySurface is an explicit specification of the u parameter value where each junction between component surfaces occurs -- the so-called "transition u values". These values must be in an ascending sequence.

269 A list-selected real entity has the wrong units for its usage.

A list-selected real entity has the wrong units. For example, this error will occur if an LSLength entity has index = 3, but the third entity in its entity list parent does not have unit dimensions of length.

270 Relabel: too few entries (minimum of 2 required).

The Relabel entity requires at least two values (0 and 1).

271 NUBFitCurve, Snake, Surf: When a KnotList is specified, the corresponding nc-specifier can't be zero.

There is a fixed relationship between the number of control points (nc), spline degree, and the number of knots. When the number of knots and degree have been specified, there is no room for nc to increase in order to improve the fit.

272 Arc types 4,5: Coincident points

The type "Arc: tangent at point1 end" and type "Arc: tangent at point3 end" arcs are specified by three points. If the first two of these points are identical, there is insufficient information to orient the arc in space.

273 Type 4 or 5 arc: Points lie on a line

The type "Arc: tangent at point1 end" and type "Arc: tangent at point3 end" arcs are specified by three points. If point2 and point3 lie on a line through point1, but in opposite directions, the construction implies a circle of infinite radius and indefinite orientation.

274 TabPoint: Too few points in specified 3DA file; unexpected EOF.

For a Tabulated Point, the valid range for Point no. is 1 to N, where N is the number of points in the .3DA file.

275 TabCurve: Too few polylines in specifed 3DA file; unexpected EOF.

For a Tabulated Curve, the valid range for Polyline no. is 1 to N, where N is the number of polylines in the .3DA file. (N is also the number of points with pen = 0.)

276 TabCurve: Too many divisions

For a Tabulated Curve, the number of points read from the .3DA file is divisions x subdivisions + 1. This error occurs when the polyline in the file does not have this many points.

277 TabSurf: Insufficient patches in the specified file.

For a Tabulated Surface, the valid range for Patch no. is 1 to N, where N is the number of patches in the .PAT file.

278 TabSurf: Wrong number of points in patch.

For a Tabulated Surface a patch needs to contain (u-divsions x u-subdivisions + 1) polylines, each consisting of (v-divisions x v-subdivisions + 1) points.

279 OffsetPt, OffsetCurve: Located at coordinate singularity; indeterminate normal.

The normal to the surface could not be evaluated at this location.

280 TanPoint: Located at zero-velocity point on curve; indeterminate direction.

The supporting bead is at a location where the curve has zero velocity; the unit tangent vector could not be evaluated.

281 Frame3: Points 1 and 2 coincide; direction is indeterminate

The three points specifying a 3 Point Frame entity must all be distinct.

282 Frame3: Points 1 and 3 coincide; direction is indeterminate

The three points specifying a 3 Point Frame entity must all be distinct.

283 Frame3: Points 1, 2 and 3 lie on a line; indeterminate normal direction

The third point specifying a frame must lie off the line from point1 to point2, or there is insufficient information to fix the rotation of the frame.

284 Entity is in error because one of its parents is in error

An entity cannot be correctly evaluated if one of its parent entities is in error. This is true even if the parent just has invalid color or visibility. This error will disappear when the parent entity is corrected.

285 WireFrame: Specified file could not be opened - wrong filename, directory, or file does not exist

SurfaceWorks was unable to locate and open the specified file. Check the filename (and optional extension) given in the entity data. Be sure the file is in the correct directory.

286 WireFrame: Error reading data from file

The file specified to contain the data for this entity does not conform to the .3DA or .PAT format. You can use Windows Notepad or another text editor to examine the file contents. For information about file formats, see Appendix C: Output File Specifications

287 Foil: Could not open file TYPExxx.FOI - it may be in the wrong directory, or it may not exist.

This foil curve has a type xxx greater than 100. This requires taking data from a file named TYPExxx.FOI. This file was not found in the specified directory. Browse to find the file, or move it to the correct directory.

288 Foil: Error reading file TYPExxx.FOI - File does not follow expected format.

This foil curve has a type xxx greater than 100. This requires taking data from a file named TYPExxx.FOI, which has a specific format. Review the contents of the .FOI file and compare it with the specification. Be sure data items are separated by spaces, not tabs.

289 One or more supports are the *EMPTY* placeholder entity

EMPTY is used to initialize a list of supports during entity creation, but it does not represent a valid entity. Select a valid entity as the support.

290 PolySurf: First patch orientation couldn't be classified.

291 PolySurf: Second or later patch orientation couldn't be classified.

292 IntRing2: snake1 and snake2 are on different surfaces.

The two Snake parents of a Snake-Snake Intersection Ring have to be on the same host surface.

293 The parameter values of the Relabel do not meet monotonicity requirements.

This error can be caused by having a decreasing sequence of parameter values or too many identical values, e.g. more than 2 values the same for degree-2, more than 3 values the same for degree-3, etc. For more information please see the User's Guide under Relabel.

294 GeoSnake: Singular equations; no solution.

A Geodesic Snake requires solution of simultaneous linear equations; the solution failed with singular equations.

295 GeoSnake: Failed to converge.

In general, a Geodesic Snake requires an iterative solution. The iteration failed to converge within the specified tolerance.

296 BSubCurve, SubCurve: At least one bead must identify an actual curve.

SubCurve and SubSnake. You cannot pick only *0 and *1 for parents. At least one parent must be an actual bead or ring on the curve or snake.

297 BSubSnake, SubSnake: At least one ring must identify an actual snake

When the system beads *0 and *1 are used as parent entities, they must be accompanied by at least one real bead or ring that identifies the host snake; otherwise the host snake is undetermined.

298 BSubCurve: A minimum of two bead supports is required.

Self explanatory, but also obsolete -- this error no longer occurs.

299 ProcCurve/Snake/Surf: Bead/Ring parents must be Absolute Beads or Rings.

300 BFitSurf: Too few u-control points for the specified B-spline type.

Geometry Errors 301 - 350

301 BFitSurf: Too few v-control points for the specified B-spline type.

302 BFitSurf: Fitting failed to meet specified tolerance.

303 ProcCurve, ProcSnake, ProcSurf: Construction failed for at least one bead position.

304 BFitSurf: Insufficient data in surface table for u direction.

305 BFitSurf: Insufficient data in surface table for v direction.

306 BFitCurve: Too few control points for the specified B-spline type.

307 BFitCurve: Insufficient data in curve table; increase divisions of basis curve.

308 BFitCurve: Fitting failed to meet specified tolerance.

309 CvContour: The curve has a zero tangent; normal plane is indeterminate.

*/

310 WireFrame: Specified file is empty (zero length)

The .3DA or .PAT file that has been named contains no data, therefore nothing can be displayed. Make a new file!

311 Procedural entity has more parent Graphs than the degrees of freedom of the moving entity.

312 One or more parents are undefined.

313 IntRing2: Failed to find a self-intersection.

314 BlendPoint, type-2: The weights sum to zero; would cause division by zero.

315 ProcPtSurf: All parent Magnets must be Absolute Magnets.

*/

316 TrimSurf: One or more snakes are on a different surface.

All snakes must be on the same host surface.

317 TrimSurf: Snakes don't connect up end-to-end to make closed loops.

Snakes must connect up end-to-end, in the specified sequence.

318 NURBSurf: The knotlist specified for the u-direction has too few knots.

A NURBS surface has too few knots for its degree and number of control points in the u-direction. If N is the number of control points and D is the polynomial degree (e.g., 3 for cubic), then a KnotList needs exactly N-D+1 entries, the first being 0 and the last being 1. A KnotList2 needs exactly N+D+1 entries, the (D+1)-th being 0 and the (D+N-1)-th being 1. Example: if N=5 and D=2, a KnotList needs 4 entries such as { 0 1/3 2/3 1 }; a KnotList needs 8 entries such as { 0 0 0 1/3 2/3 1 1 1 }.

319 NURBSurf: The knotlist specified for the u-direction has too many knots.

320 NURBSurf: The knotlist specified for the v-direction has too few knots.

321 NURBSurf: The knotlist specified for the v-direction has too many knots.

322 IntSnake: Cutting surface is insufficiently smooth for non-zero offset

The surface used as the support for the Intersection Snake is not continuous. This can happen, for instance, with an Offset Surface made from a PolySurface.

323 ArcLenBead, Ring: Attempt to differentiate arc length at a zero-velocity point.

An Arc-length Bead or Ring may have difficulty locating itself of its supporting curve has zero-velocity points or too radical a velocity distribution.

324 ArcLenBead, Ring: Search for arc distance failed to converge.

An Arc-length Bead or Ring generally requires an iterative solution. The iteration failed to converge to a satisfactory tolerance. Check that you have specified the intended host curve. Use a bead near the desired location for a better starting value.

325 TrimSurf cannot serve as support for this entity.

A TrimSurf cannot serve as the support for an OffsetSurf, MirrorSurf, CopySurf, etc. Workaround: Perform the intended operation (offset, mirror, copy, etc.) on the TrimSurf's base surface, and make a new TrimSurf with CopySnakes for boundaries.

326 SubSnake: The host of supporting beads must be a snake.

SubSnake's parent rings (or beads) must have snake as parent.

327 The host surface cannot be evaluated at the requested location.

328 TrimSurf: A boundary loop crosses itself.

Each boundary loop must be a closed loop, which does not intersect itself.

329 TrimSurf: Two or more boundary loops cross each other.

A TrimSurf has one outer boundary and zero or more inner boundaries. The boundary loops must not cross themselves, or any other boundary loops of this TrimSurf.

330 TrimSurf: A trimming loop has zero area.

A trimming loop is required to enclose some area. This error can occur when all the snakes of a loop have zero length, or lie on top of each other.

331 The host curve cannot be evaluated at the requested location.

332 The host snake cannot be evaluated at the requested location.

333 TrimSurf: A trimming loop is outside the outer loop.

A TrimSurf has one outer boundary, plus zero or more inner boundary loops representing holes. The outer boundary has to be the first loop in the set of trimming snakes.

334 TrimSurf: A trimming loop is inside an inner loop.

A TrimSurf has one outer boundary, plus zero or more inner boundary loops representing holes. Inner boundaries can't be inside each other.

335 The host solid cannot be evaluated at the requested location.

336 TrimSurf: Triangulation failed.

Check that the outer and inner trimming loops are in an appropriate order. Try changing the number of triangles.

337 PolyGraph labeling error; transition t values must be strictly increasing.

338 A specified tolerance was not met.

339 The entity type is not supported by this program version.

340 The entity failed to evaluate as needed.

341 Point entities are not allowed as parents for BFitCurves.

342 BlendSurf2: A supporting entity does not qualify to establish a specified tangency.

Each boundary curve has to be either (1) a snake ring or magnet, (2) a planar curve, or (3) a point, in order to establish a tangency condition on that edge.

343 BlendSurf2: Normal direction could not be evaluated from a supporting entity.

Each boundary curve has to supply a normal direction in order to support a tangency condition on that edge.

344 BSubSnake: A minimum of 2 bead or ring supports is required.

Self explanatory -- also obsolete, not treated as an error anymore.

345 SolveSet: general error.

346 PolyCurve2: Unable to assign transition t values to balance velocities at knots.

347 CenterPoint cannot be located for a Line, a straight curve, or at an inflection.

348 CenterPoint cannot be located at a point where the curve has zero velocity.

349 RadiusArc, type 1 or 2: 3 points are on a line; plane is indeterminate.

350 RadiusArc, type 1 or 2: Points 1 and 3 are more than (2 x radius) apart.

Geometry Errors 351 - 400

351 IntBead2 failed to meet tolerance.

352 Solid has zero u-divisions or subdivisions.

353 Solid has zero v-divisions or subdivisions.

354 Solid has zero w-divisions or subdivisions.

355 The default graph * cannot be used as a component of a Polygraph.

*/

356 A ProjSnake2 with non-zero draft angle requires a plane mirror.

If you want to create a Projected Snake that has a non-zero draft angle, you must use a plane for its Mirror/Surface.

357 ProjSnake2 with non-zero draft angle: projection direction failed.

This error occurs when the tangent of the curve being projected is normal to the plane of projection, so a suitable frame cannot be constructed.

358 SuperBlend has the wrong number of control values; should be (ncu-2) x (ncv-2).

359 BRelCurve: With 3 or more control points, the only valid Graph is the default graph (*).

360 Ring or Magnet used as trimming curve.

A ring or magnet is ineligible for service as a trimming curve. You can just leave it out of the trimming loop, and get the same effect.

361 A trimming curve has zero length.

Each snake used in a trimming loop must have non-zero length. If a snake has zero length, you can just leave it out of the boundary.

362 The range of a Variable has high value less than low value.

363 The value of the Variable is set below its specified range.

364 The value of the Variable is set above its specified range.

365 Operand in formula is not a real valued entity or real constant.

366 Operand of unary '+' must be a real.

367 Operand of unary '-' must be a real.

368 Operands of binary '+' must be reals.

369 Operands of binary '-' must be reals.

370 Operands of '*' must be reals.

371 Operands of '/' must be reals.

372 Operands of '^' must be reals.

373 The argument of SIN or SIND must be a real.

374 The argument of COS or COSD must be a real.

375 The argument of TAN or TAND must be a real.

376 The argument of ATN or ATND must be a real.

377 The arguments of ATN2 or ATN2D must be reals.

378 The argument of LOG or LOG10 must be a real.

379 The argument of EXP must be a real.

380 The argument of ABS must be a real.

381 The argument of SQRT must be a real.

382 The argument of ROUND must be a real.

383 The arguments of MIN must be reals.

384 The arguments of MAX must be reals.

385 The argument of TPOS must be a bead or ring.

386 The argument of UPOS or VPOS must be a magnet or ring.

387 The argument of XPOS, YPOS or ZPOS must be a point.

388 The arguments of DIST must be points.

389 Formula: The first argument of CLEAR does not qualify for clearance.

390 The arguments of ANGLE must be three points.

391 The argument of SIN or SIND must be unitless.

392 The argument of COS or COSD must be unitless.

393 The argument of TAN or TAND must be unitless.

394 The argument of ATN or ATND must be unitless.

395 The arguments of ATN2 or ATN2D must have the same units.

396 The argument of LOG or LOG10 must be unitless.

397 The argument of EXP must be unitless.

398 The argument of SQRT must have even unit dimensions.

399 Both arguments of MIN must have the same units.

400 Both arguments of MAX must have the same units.

Geometry Errors 401 - 450

401 Division by zero in formula evaluation.

402 Underflow or overflow during arithmetic operations.

403 Negative power of 0: x^y with x evaluating to 0 and y to a negative number.

404 Non-integral power of negative number: x^y with x evaluating to a negative number and y to a non-integral number.

405 An exponentiation (x^y) has operand(s) out of computational range.

409 Both arguments of ATN2 or ATN2D evaluate to zero.

410 Argument of LOG or LOG10 does not evaluate to a strictly positive number.

411 Argument of EXP has too large a value, out of computational range.

412 Argument of SQRT is negative.

413 At least two arguments of ANGLE are points with coincident coordinates.

414 The units of the result of the evaluation of a formula differ from the units specified in the definition of the formula.

415 Formula evaluation results in non-integer unit dimensions.

416 Entity has real parents of the wrong units.

417 A real used as an exponent must be unitless.

418 The components of this polycurve or polysnake do not join end-to-end.

PolyCurve and PolySnake parents must join end-to-end.

419 EmbedSurf: The first solid parent is required to be a BlockSolid.

420 EmbedSurf: Singular equations, or iteration failed to converge.

421 EmbedSurf: One of the three Relabels of the BlockSolid parent cannot be inverted.

*/

422 BSubCurve or Snake: The supporting beads or rings must be in either a non-decreasing or a non-increasing sequence

When making a SubCurve or SubSnake the beads or rings must be selected in the order in which they exist on the parent curve or snake.

423 Zero-degree continuity is required between the component graphs of a PolyGraph.

424 The parent surfaces are not all connected or are not all connected by edge adjacency.

425 The composite surface is not closed and hence does not make a shell.

426 The parent surfaces are not connected so as to make a solid, that is an outer shell containing zero or more disjoint inner shells.

427 Composite surface not orientable.

428 Shell cannot be oriented.

429 B-rep solid cannot be oriented because either its outer shell or one of its inner shells is not orientable.

430 IntSnake: failed to converge.

An Intersection snake is computed by a series of nonlinear iterations. At one or more points along the snake, the iteration failed to converge.

431 IntSnake: singular equations.

An Intersection snake is computed by solution of nonlinear equations. At one or more points along the snake, the governing equations were singular, i.e. had no definite solution. The usual cause of this error is nonexistent or grazing intersections.

432 IntSnake: failed to pass through a coordinate singularity on the host surface.

A coordinate singularity is a place where the surface has zero velocity with respect to one or both parameters -- for example, a pole --, or the u and v directions are the same.

433 Boundary Solid: wrong number of surfaces (6 required).

434 Relabel: first value not equal to 0, last value not equal to 1 and/or values outside 0 to 1.

435 ProjSnake2: Parent curve is insufficiently smooth for non-zero draft angle.

436 BSolid: ncu, ncv, ncw must each be at least two.

437 BSolid: number of control points not evenly divisible by ncu x ncv.

438 ProxBead, Ring or Magnet: Failed to converge.

439 A parent could not be evaluated at the requested location (usually outside the parameter interval 0 to 1).

440 Formula: First argument of ARCLEN is not a curve

441 Formula: Arguments 2 and 3 of ARCLEN do not have correct units (unitless).

442 Contour Curve: Index is outside the index range of the contours entity.

443 Contour Curve: Contour is closed, can't determine starting point.

444 **Contour Curve: Contour cuts could not be assembled into a curve.**

445 **Contour Curve: Assembled contour can't be oriented (distance to mirror failed).**

446 **Contour Curve: Contour cuts do not join up within tolerance.**

447 **Parent surface is not sufficiently smooth.**

448 **Cannot evaluate normal at some point on parent surface.**

449 **TriMesh breaks data is bad.**

450 **TriMesh breaks data has an out-of-range node index.**

Geometry Errors 451 - 500

451 **The index of a node in the triangle data is out of range.**

452 **One or more triangles are degenerate (area too small).**

453 **The mesh is disconnected.**

454 **More than 2 triangles sharing an edge.**

455 **One or more triangles are duplicates.**

456 **The triangle data contains one or more nodes not used in any triangle.**

457 **TriMesh Magnet: The triangle index is out of range.**

458 **Could not construct triangle mesh from the given data.**

459 **Expanded Surface: Flattening failed to converge.**

460 **Expanded Surface: Flattening equations were singular.**

461 **Expanded Surface: Flattening failed at coordinate singularity.**

462 **Control Point: The parent curve or surface is not NURBS-exact.**

463 **Control Point: First index is out of range.**

464 **Control Point: Second index is out of range.**

465 **Break Bead, Break Ring: max-degree parameter is out of range.**

466 **Break Bead, Break Ring: breakpoint index is out of range.**

467 **TMEdgeSnake: One or both magnets are off the Trimesh boundary.**

468 **TMEdgeSnake: The magnets are on different boundary loops.**

469 **The cut specified by index has no intersection points.**

470 **Rolling Ball Fillet: Zero velocity point on curve parent; normal plane indeterminate.**

471 **Rolling Ball Fillet: Construction failed.**

472 **Rolling Ball Fillet: The curve is not sufficiently smooth to serve as a path.**

473 **The tangent direction could not be determined at the ring support.**

474 **CENTROID function: Argument 3 (XYZ index) is out of range (1 to 3).**

475 **AREA, VOLUME, CENTROID functions: Argument 2 (use symmetry) is out of range (0-1).**

476 FRAMEPOS function: Argument 3 (xyz index) is out of range (1 to 3).

477 The boundary representation is non-manifold.

478 PolySurf has discontinuities between its component surfaces.

479 TabReal: Row-index or column-index is negative.

480 TabReal: The value as specified was not found in the file.

481 TabReal: The row-index exceeds the number of lines in the file.

482 TabReal: The column-index exceeds the number of tokens in the specified row.

483 TabGraph: Abscissa values must be an ascending sequence.

484 TabGraph: column-index1 and column-index2 must both be positive.

485 TabGraph is declared periodic, but first and last values don't match.

486 Argument of the SIGN function must be a real.

487 This alarm function is tripped (set, on an enabled layer, and second argument < 0).

488 PolyGraph: Component graphs have different unit dimensions.

489 Composite Surface or B-Rep Solid: One or more of the surface supports are duplicates.

490 Radius Arc, type 4: No intersection was found between the arc and the Point2-Point3 line.

491 Tabulated Point, Curve, Snake, or Surface: Index must round to a positive number.

492 Control Point: An index value must round to a positive number.

493 Break Bead or Break Ring: Index must round to a positive number.

494 TabGraph: The specified starting abscissa value was not found in the file.

495 TabGraph: The specified ending abscissa value was not found in the file.

496 TrimSurf2: The magnet parent is on one of the boundary snakes, or on the base surface boundary.

497 TrimSurf2: The magnet parent is inside two or more boundary loops.

498 Formula: The second argument of CLEAR does not qualify for clearance.

499 Procedural Curve or Snake: The construction would produce a 0-degree breakpoint (a tear or discontinuity) in the curve or snake.

500 Procedural Surface: The construction would produce a 0-degree breakline (a tear or discontinuity) in the surface.

Geometry Errors 501 -

501 Color: An RGBA value is out of range (0 to 1).

502 A point could not be evaluated on a snake serving as a trimesh snake.

503 ExpdTriMesh: The two Triangle Mesh Magnet parents are at the same location.

504 ExpdTriMesh: The flattening solution failed to converge.

505 ExpdTriMesh: The Triangle Mesh entity to be flattened is a closed body.

506 ExpdTriMesh: The Triangle Mesh entity to be flattened has one or more holes.

507 ExpdTriMesh: The Triangle Mesh entity to be flattened is of the wrong topological genus.

508 TMCopyMagnet, TMCopySnake: The source and host triangle meshes have different triangulations.

509 Expanded Surface: The two magnet parents have different host surfaces.

510 Expanded Surface: The two magnet parents are at the same location, so do not provide a direction to orient the expansion.

511 Supports of an entity array have different array sizes.

512 Strain Contours: One or more surface parents are not types that support strain contours.

513 CopyContours: Source is not in Contours list of surfaces to cut.

514 CopyContours: Destination class vs. Source class is not a supported combination.

515 CopyContours: First or last index is out of range.

516 CopyContours: Destination trimesh is not congruent to Source.

517 BlendPoint, types 0 and 1: All weights must be unitless.

518 BlendPoint, type 2: All weights that are real parents must have the same unit dimensions.

519 A magnet or ring serving as a trimesh magnet could not be evaluated as a trimesh magnet.

520 A snake serving as a trimesh snake could not be evaluated as a trimesh snake.

521 A surface or Composite Surface failed to make a valid triangulation, to serve as a triangle mesh.

522 RealList: An element that is a real parent has the wrong unit dimensions.

523 ForRealList: The starting value is a real parent that doesn't have the correct unit dimensions.

524 ForRealList: The increment is a real parent that doesn't have the correct unit dimensions.

525 ForRealList count, if a real parent, must be unitless.

526 STRAIN function: Argument 2 (index) is out of range (0 to 1).

527 At least one magnet parent must identify an actual host surface.

528 CURV function: Argument 3 (kind) is out of range (0 to 2).

529 CURV function: for kinds 1 (normal) and 2 (geodesic), the host curve must be a snake.

530 CURV function can't be evaluated where velocity is zero on the host curve or snake.

Index

B

E

F

G

M

O

P

Q

R

S

T

U

V

W