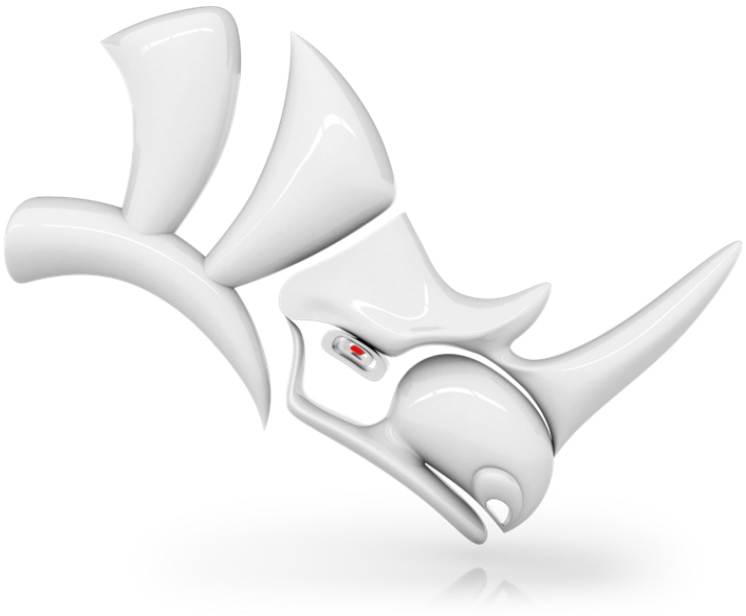
Rhinoceros®

modeling tools for designers

Training Manual

Level 2



RH60-TM-L2 Revised: 10/27/2020

Rhinoceros Level 2 Training Guide

Printed in USA

Content Credit:

Pascal Golay, Robert McNeel & Associates

Mary Ann Fugier, Robert McNeel & Associates

Jerry Hambly, Robert McNeel & Associates

Vanessa Steeg, Robert McNeel & Associates

Corrections or additions: please email Mary Ann Fugier mary@mcneel.com.

Proofing Credit:

Bob Koll, Robert McNeel & Associates

Lambertus Oosterveen

Vanessa Steeg, Robert McNeel & Associates

Table of Contents

Table of Contents ... iii
Chapter 1 - Introduction ... **7**
Software ... 7
Target audience ... 7
Duration ... 7
Prerequisites ... 7
Course objectives ... 7
Three classroom days ... 8
Six half days (on-line training) ... 9
Chapter 2 - Getting the Models ... **11**
Exercise 2-1 Setting up your workstation ... 11
Option 1: Download files separately ... 11
Option 2: Download all files at once ... 11
Exercise 2-2 Warm-up ... 12
Exercise 2-3 Make a trackball mouse ... 12
Chapter 3 - Customizing the Rhino UI ... **13**
The toolbar layout ... 13
Exercise 3-1 Customizing Rhino's Interface ... 13
Rules for commands in buttons ... 20
Command aliases ... 24
Macro editor ... 25
Export and import Rhino options ... 25
Shortcut keys ... 25
Plug-ins ... 26
Scripting ... 29
Template files ... 31
Exercise 3-2 Create a template ... 31
Chapter 4 - NURBS topology ... **35**
Exercise 4-1 Work with topology ... 35
Exercise 4-2 Observe trimmed surfaces ... 39
Custom display modes ... 41
Exercise 4-3 Color the backface ... 41
Chapter 5 - Curve creation and continuity ... **45**
Curve degree ... 45
Exercise 5-1 Observe curve degree ... 45
Curve and surface continuity ... 47
Not continuous ... 47
Positional continuity (G0) ... 47
Tangency continuity (G1) ... 47
Curvature continuity (G2) ... 48
Curve continuity and curvature graph ... 48
Exercise 5-2 Examine geometric continuity ... 53
Set up aliases ... 54
Exercise 5-3 Make Along and Between aliases ... 54
Tangent continuity ... 55
Tab direction lock ... 55
Curvature continuity ... 60
Exercise 5-4 Match the curves ... 60

Advanced techniques for controlling continuity ... 61
Chapter 6 - Surface continuity ... 63
Analyze surface continuity ... 63
Match surface continuity ... 63
Match surface options ... 63
Isocurve direction adjustment ... 63
Surface continuity and MatchSrf ... 63
Exercise 6-1 Practice matching surface continuity ... 63
Add knots to control surface matching ... 67
Use EndBulge to edit the surface shape ... 68
Match surfaces ... 69
Surfacing commands that pay attention to continuity ... 71
Exercise 6-2 Create a surface from a network of curves ... 71
Create a patch surface ... 73
Patch options ... 75
Exercise 6-3 Make a patch from an edge and points ... 75
Lofting ... 76
Exercise 6-4 Make a lofted surface ... 76
Blends ... 77
Exercise 6-5 Make a surface blend (BlendSrf 1) ... 78
Surface blend options ... 86
Exercise 6-6 Make a surface blend with options ... 86
Fillets, blends, and corners ... 89
Exercise 6-7 Make a corner fillet with three different radii ... 89
Exercise 6-8 Make a variable radius blend ... 90
Exercise 6-9 Make a six-way fillet using a patch ... 91
Chapter 7 - Modeling with history ... 93
Activating history ... 94
Why is history off by default? ... 94
Steps in the history chain ... 95
History-enabled commands ... 96
History-enabled commands ... 96
Advanced Surfacing Strategies ... 97
Exercise 7-1 Soft corners (part 1) ... 98
Soft corners - another method ... 99
Exercise 7-2 Soft corners (Part 2) ... 99
Chapter 8 - Advanced surfacing concepts ... 105
Dome-shaped buttons ... 105
Exercise 8-1 Soft domed buttons ... 105
Creased surfaces ... 114
Exercise 8-2 Surfaces with a crease (part 1) ... 115
Surfaces with a crease - Part 2 ... 119
Exercise 8-3 Surfaces with a crease (Part 2) ... 119
Input curve fairing to control surface quality ... 122
Chapter 9 - Modeling from reference images ... 129
Exercise 9-1 Handset ... 129
Chapter 10 - An approach to modeling ... 145
The Cutout ... 145
Exercise 10-1 Set up and build from the "floor" surface ... 145
Build the sides of the cut-out ... 158

Transition surfaces ... 160
Chapter 11 - Applying 2-D graphics ... 167
Exercise 11-1 Importing an Adobe Illustrator file ... 167
Flow the logo onto a free-form surface with history ... 172
Make a model from a 2-D drawing ... 176
Exercise 11-2 Making the detergent bottle ... 177
Chapter 12 - Surface analysis ... 183
Exercise 12-1 Surface analysis ... 183
Chapter 13 - Sculpting ... 189
Tools to help in control point editing ... 189
Gumball ... 189
DragMode ... 189
Nudge ... 190
SetPt ... 190
InsertKnot ... 190
Some considerations when inserting knots ... 190
InsertControlPoint ... 190
Exercise 13-1 Dashboard ... 191
Another way to tweak the shape ... 193
Add knots ... 193
Add details ... 194
Chapter 14 - Deformation tools ... 197
Deforming objects ... 197
CageEdit ... 197
Exercise 14-1 Using cage editing to deform an object ... 197
Exercise 14-2 CageEdit the salad fork ... 198
Using other deformation tools ... 201
Stretch ... 201
Exercise 14-3 Stretch an object ... 201
Orient an object on a surface ... 202
Exercise 14-4 Place a small detail on an object ... 202
Deform an object in a spiral ... 203
Exercise 14-5 Deform using Maelstrom ... 203
Flow along a curve ... 204
Exercise 14-6 Deform an object by flowing it along a curve ... 204
Flow ... 206
Making a Ring with Flow ... 206
Exercise 14-7 Flow the parts of a ring along the shank curve ... 206
Chapter 15 - Blocks ... 213
Instances and definitions ... 213
Defining blocks ... 213
Insertion points ... 213
Embedded and linked blocks ... 213
Layers and blocks ... 213
The Block Rules ... 213
Blocks ... 214
Exercise 15-1 Block basics ... 214
Files as blocks ... 216
Exercise 15-2 Inserting files as block ... 216

Chapter 16 - Troubleshooting **219**
General strategy 219
Start with a clean file 219
Exercise 16-1 To try these procedures 220
Chapter 17 - Polygon meshes **223**
Render meshes 223
Meshes for manufacturing 223
Exercise 17-1 Experiment with mesh settings 224
Meshes from NURBS objects 225
Chapter 18 - Rendering **229**
Exercise 18-1 Rhino rendering 229
Rendering properties 233
Exercise 18-2 Rendering with Environments 233
Scene lighting 236
Image and bump maps 239
Decals 239
Chapter 19 - Introduction to Grasshopper **246**
The Bike Wheel 246
The Grasshopper Canvas 246
The Grasshopper Settings 246
The Finder 247
Create the Circles 248
Divide the Circle 250
Connect the Points 250
Pipe the Curves 251
Orienting the Wheel 252
Mirror Front Wheel 253
Scale the Front Wheel 254
List Item to Select the Tire 256
Locating the Bottom Surface of the Wheel Bounding Box 257
Scaling the Front Bike Wheel from Bottom 258
Bake The Wheels 258

Chapter 1 - Introduction

The course explores advanced techniques in modeling to help participants better understand Rhino's modeling tools, advanced surfacing commands, curves and surface topology and how to apply these concepts in the in practical situations.

In class, you will receive information at an accelerated pace. For best results, practice at a Rhino workstation between class sessions, and consult the **Rhino Help** system from the **Help** menu: Help Topics.

Software

The training guide was designed to be used with Rhinoceros 6 or later.
The training files have been updated to open with Rhinoceros 6 or later.

Target audience

This course and training guide were compiled to accompany the Rhinoceros Level 2 instructor-lead training sessions. This course is designed for individuals who will be using Rhino's advanced features or supporting Rhino.

Duration

Typically this course is presented over three - 8 hour days of training for a total of 24 hours.

The training can be presented in three full-day, six half-day sessions or adapted for a custom schedule.

Instructor should prepare by choosing which exercises are to be presented during class and which exercise work will be assigned as homework.

Prerequisites

Completion of Level 1 training or equivalent, plus three months (minimum) experience using Rhino.

Course objectives

In Level 2, you learn how to:

- Customize toolbars and toolbar collections
- Create simple macros
- Use advanced object snaps
- Use distance and angle constraints with object snaps
- Construct and modify curves that will be used in surface building using control point editing methods
- Evaluate curves using the curvature graph
- Use a range of strategies to build surfaces
- Rebuild surfaces and curves
- Control surface curvature continuity
- Create, manipulate, save and restore custom construction planes
- Create surfaces and features using custom construction planes
- Group objects
- Visualize, evaluate, and analyze models utilizing shading features
- Place text around an object or on a surface
- Map planar curves to a surface
- Create 3-D models from 2-D drawings and scanned images
- Clean up imported files and export clean files
- Use rendering tools

Three classroom days

Day 1	Topic
8:00 am – 9:30 am	Introduction and warm up exercise
9:30 am – 12:00 pm	Interface and customization
12:00 pm – 1:00 pm	Lunch
1:00 pm – 3:00 pm	NURBS topology and curve degree
3:00 pm – 5:00 pm	Curve and surface continuity
Day 2	**Topic**
8:00 am – 10:00 am	History, advanced surfacing and construction plane tools
10:00 am – 12:00 pm	More construction planes, mapping objects to surfaces
12:00 pm – 1:00 pm	Lunch
1:00 pm – 3:00 pm	Surface analysis
3:00 pm – 5:00 pm	Putting it all together—Scoop exercise
Day 3	**Topic**
8:00 am – 10:00 am	More construction planes, mapping objects to surfaces
10:00 am – 12:00 pm	Surface analysis, direct surface manipulation
12:00 pm – 1:00 pm	Lunch
1:00 pm – 3:00 pm	Blocks, troubleshooting, meshing
3:00 pm – 5:00 pm	Rendering (time allowing)
	Class schedules are suggested. Actual class schedule will be set by the instructor.

Six half days (on-line training)

Session 1	Topic
9:00 am – 10:45 am	Introduction and warm up exercise
11:00 am – 12:30 pm	Interface and customization
Session 2	**Topic**
9:00 am – 10:45 am	NURBS topology and curve degree
11:00 am – 12:45 pm	Curve and surface continuity
Session 3	**Topic**
9:00 am – 10:45 am	History, advanced surfacing and construction plane tools
11:00 am – 12:45 pm	More construction planes, mapping objects to surfaces
Session 4	**Topic**
9:00 am– 10:45 am	Surface analysis
11:00 am – 12:45 pm	Putting it all together—Scoop exercise
Session 5	**Topic**
9:00 am – 10:45 am	More construction planes, mapping objects to surfaces
11:00 am – 12:45 pm	Surface analysis, direct surface manipulation
Session 6	**Topic**
9:00 am – 10:45 am	Blocks, troubleshooting, meshing
11:00 am – 12:45 pm	Rendering (time allowing)
Questions	12:45 pm – 1:00 pm
End of Class	1:00 pm

Chapter 2 - Getting the Models

Exercise 2-1 Setting up your workstation

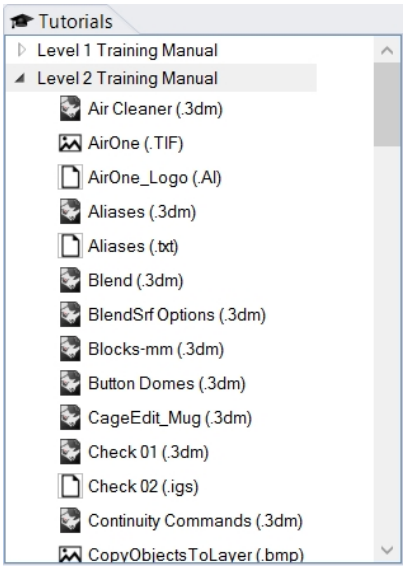
You can access the models used in this training guide with two primary options. Rhino will download each file for you as you need it or you may download all the files as a compressed zip and unzip them into a folder.

Note: You will need basic file management skills to use Rhino effectively on your computer. If you are not familiar with creating folders, copying, renaming or deleting files, stop now and find training to establish these skills.

Option 1: Download files separately

If you are new to file management on Windows, this is the better option. If you prefer to avoid all the individual file downloads, then continue with *Option 2: Download all files at once*.

1. Create a folder on your **Desktop** or in your **My Documents** folder, or another location in which you have full rights.
2. Name the folder **Level 2 Training** or other name that you will remember.
3. Open the **Rhino** application.
4. From the **Help** menu, click **Learn Rhino**, and click **Tutorials and Samples**.
   The **Tutorials** panel will appear.



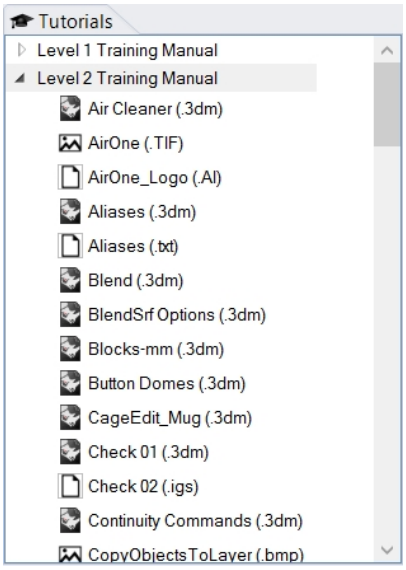
5. Navigate to the **Level 2 Training Manual** folder, and scroll to the model.
6. Double-click the file. This will load the contents of the file into a new Rhino model.
7. At the end of each section, save the file to the folder you created in the previous steps.
8. Repeat these steps at the beginning of any exercise that directs you to open an existing file.
9. For exercises that require image files, download the required files and save them into the same folder that you have saved your 3dm model file in step 7. You will need to repeat this for each image file.

   For example, rendering the Mug.3dm model requires all of these files:
   MintyGreen-Box End.png, MintyGreen-Box Side.png, MintyGreen-Box_upper.png, MintyGreen-Floss.png, MintyGreen-SideFlap_RGBA.tif, MintyGreen-TopFlap_RGBA.tif, MintyGreen-Tube.png, Sailboat_RGBA.tif

Option 2: Download all files at once

You will download a set of existing models and files that are used in this training guide.
Unzip the files in a Training folder. When you are asked to open a file, navigate to that folder.

1. Create a folder on your **Desktop** or in your **My Documents** folder, or another location in which you have full rights.
2. Name the folder **Level 2 Training** or other name that you will remember.
3. Open the **Rhino** application.
4. From the **Help** menu, click **Learn Rhino**, and click **Tutorials and Samples**.
   The **Tutorials** panel will appear.

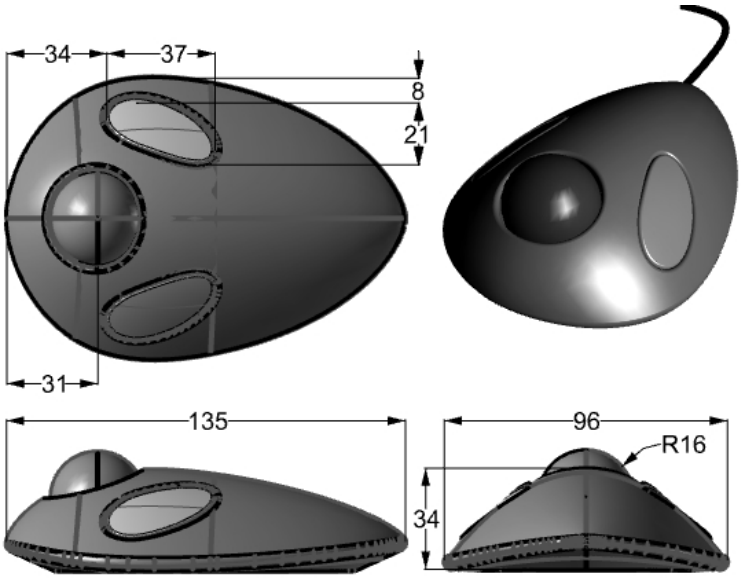


5. Navigate to the **Level 2 Training Manual** folder.
6. Double-click the **Rhino 6 Level 2 Models (.zip)** to download it.
7. **Unzip** the downloaded file into the folder that you created in the previous steps.
8. In the Rhino application, on the **File** menu, click **Open**.
9. In the **Open** dialog box, navigate to the **Level 2 Training** folder and **Open** the required model.

Exercise 2-2 Warm-up

Let's get to know you and how well you know Rhino.

Exercise 2-3 Make a trackball mouse



1. Begin a new model, save it as **Trackball.3dm**.
2. Model a trackball mouse on your own.
   The dimensions are in millimeters.
   Use the dimensions as guides only.

Chapter 3 - Customizing the Rhino UI

This chapter discusses customizing Rhino for Windows interface with the following tools:

- Toolbar layout
- Macro Editor
- Shortcut keys
- Scripting
- Template files

Note: If you are using Rhino for Mac version 6 or later, please skip this section and see this tutorial series Customizing Tool Pallets in Rhino 6 for Mac.

The toolbar layout

The toolbar layout is the arrangement of toolbars containing command buttons on the screen. The toolbar layout is stored in a file with the .rui extension that you can open and save. Rui files contain command macros, icons in three sizes, as well as tooltips and button text. Rhino comes with a default toolbar file and automatically saves the active toolbar layout before closing unless the .rui file is read-only. You can create your own custom toolbar files and save them for later use.

You can have more than one toolbar file open at a time. This allows greater flexibility to display toolbars for particular tasks.

Rhino's customization tools make it easy to create and modify toolbars and buttons. Adding to the flexibility is the ability to combine commands into macros to accomplish tasks that are more complex. In addition to toolbar customization, it is possible to set up command aliases and shortcut keys to accomplish tasks in Rhino.

Exercise 3-1 Customizing Rhino's Interface

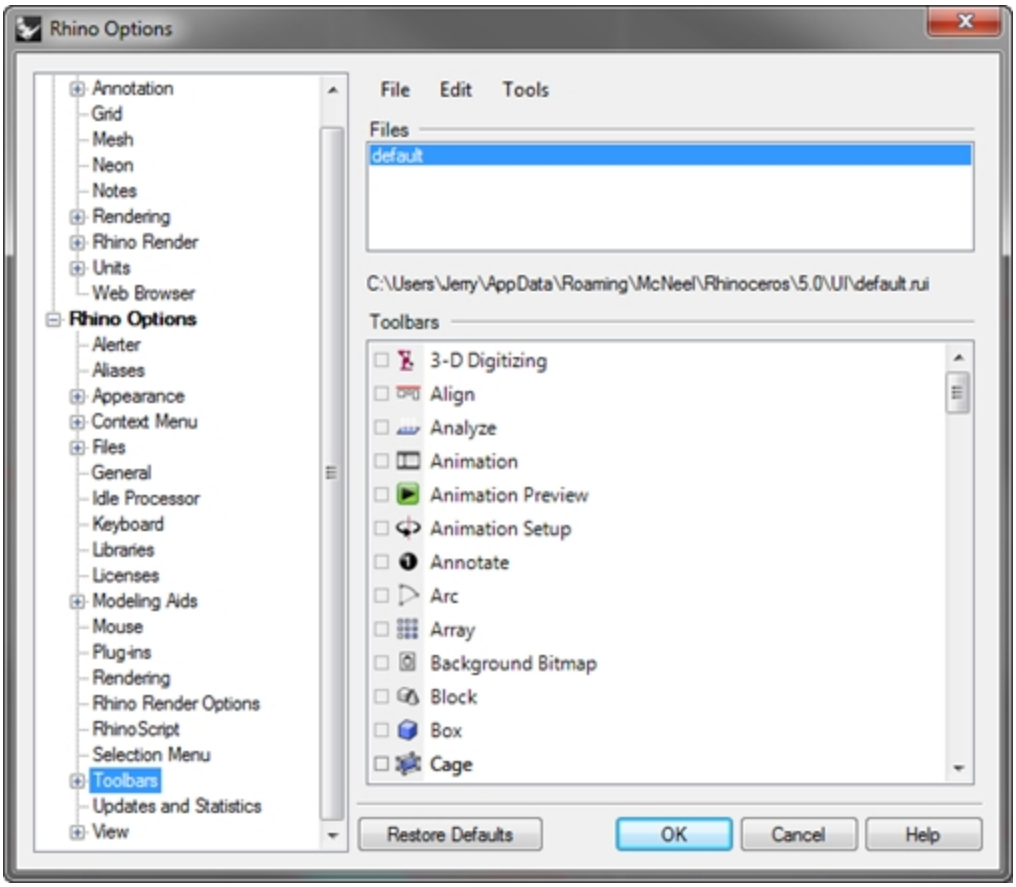
In this exercise, we will create buttons, toolbars, macros, aliases, and shortcut keys that will be available to use throughout the class.

Create a custom toolbar collection

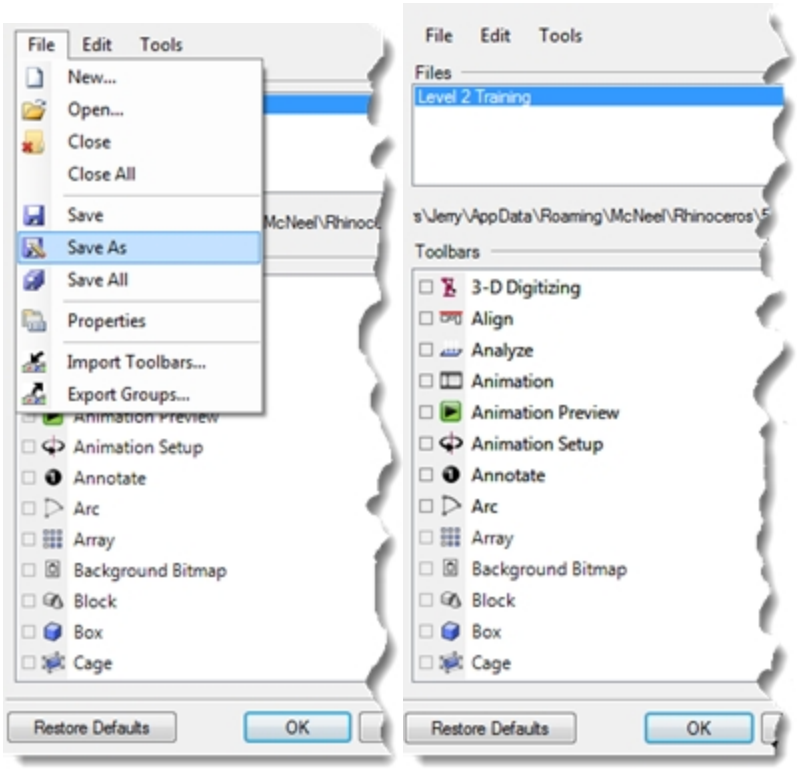
There are times that the standard commands and buttons do not do exactly what you want. For example, Zoom Extents will look at all of the objects in a model and then zoom to the extents of these objects. In this exercise, we will open a model that has several objects including some light objects.

Let us say we want to use Zoom Extents to zoom to the objects, but we do not want the command to consider the light objects. In this exercise, we will make a new toolbar with a button that will Zoom Extents while ignoring any light objects in the model.

1. **Open** the model **ZoomLights.3dm**.
2. On the **Tools** menu, click **Toolbar Layout**.
3. In **Rhino Options** dialog box, on the **Toolbars** page, choose the **default** toolbar file.

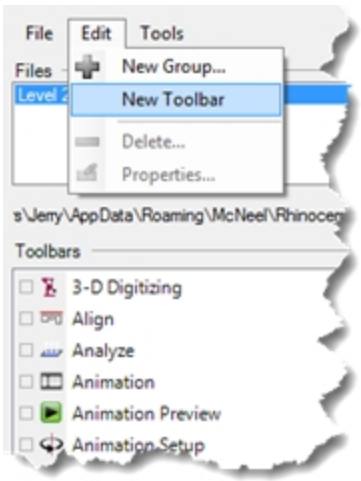


4. On the **Toolbars** page, click the **File** menu, click **Save As**.
5. In the **File name** box, type **Level 2 Training**, and click **Save**.
   A copy of the current default toolbar file is saved with the new name.
   Toolbar files are saved with a .rui extension. You will use this new toolbar file to do some customization.
   In the **Rhino Options** dialog box, on the **Toolbars** page, all the open toolbar files are listed along with a list of all the individual toolbars for the selected toolbar file.
   Check boxes show the current state of the toolbars. A checked box indicates that the toolbar is displayed.

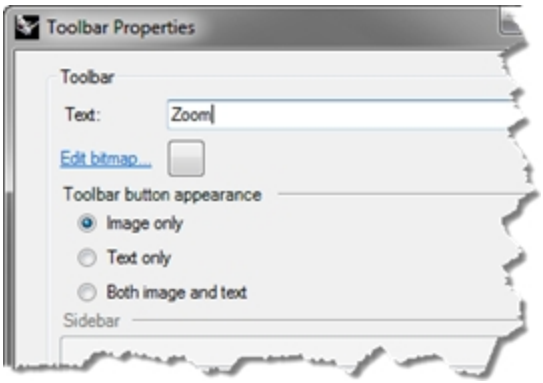


Create a new toolbar

1. On the **Toolbars** page, on the **Edit** menu, click **New Toolbar**.



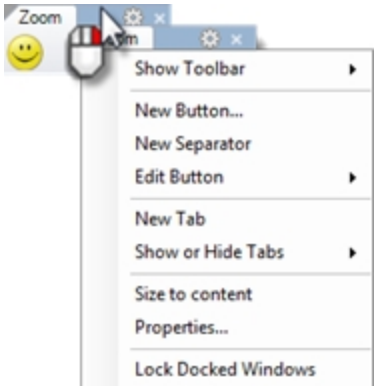
2. In the **Toolbar Properties** dialog box, in the **Text** box, type **Zoom**, and click OK.
   A new toolbar appears with a single button.



3. In the **Rhino Options** dialog box, click OK.

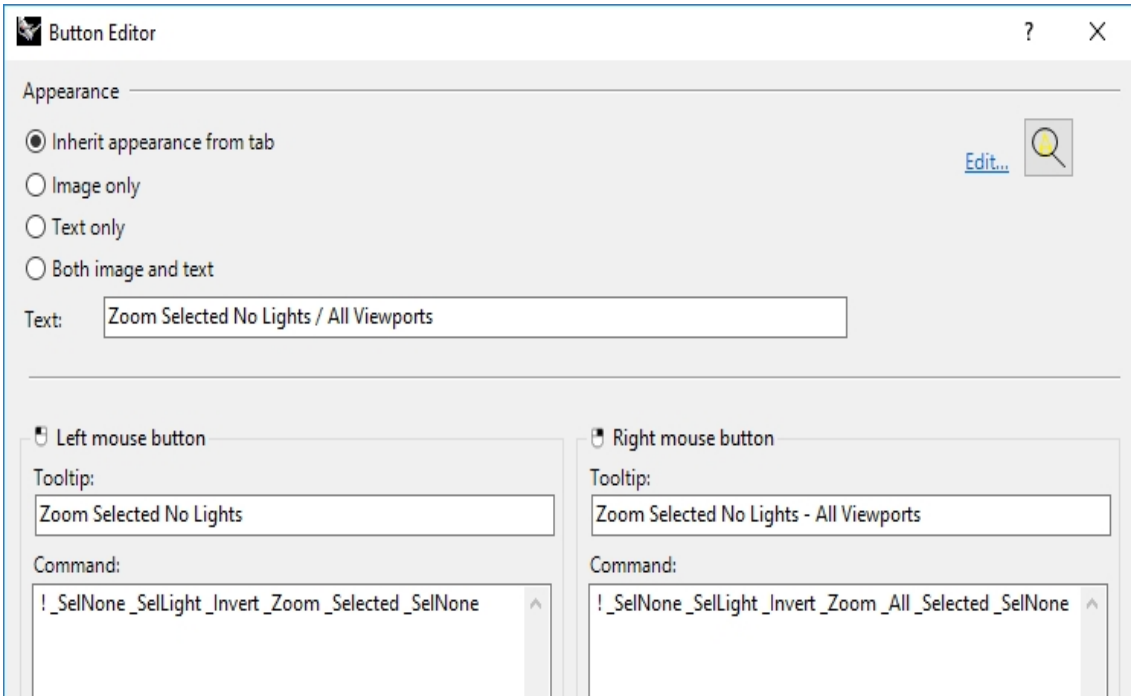
Use the title bar of a floating toolbar

- **Right-click** the **title bar** of the new toolbar you created.
  A pop-up list of toolbar options and commands displays.



Edit the new button

1. Hold down the Shift key and right-click the smiley face button in the new toolbar.
   The **Button Editor** dialog box appears with fields for commands for the left and right mouse buttons, as well as for the tooltips.
2. In the **Button Editor** dialog box, click **Image only**.
3. In the **Text** box, type **Zoom No Lights**.
4. For the **Left mouse button Tooltip**, type **Zoom Extents except lights**.
5. For the **Right mouse button Tooltip,** type **Zoom Extents except lights all viewports**.
6. In the **Left mouse button Command** box, type
   ! _SelNone _SelLight _Invert _Zoom _Selected _SelNone.
7. In the **Right mouse button Command** box, type
   ! _SelNone _SelLight _Invert _Zoom _All _Selected _SelNone.
   Note: You will find these macros in a Level 2 materials folder in text file **Macros.txt**.

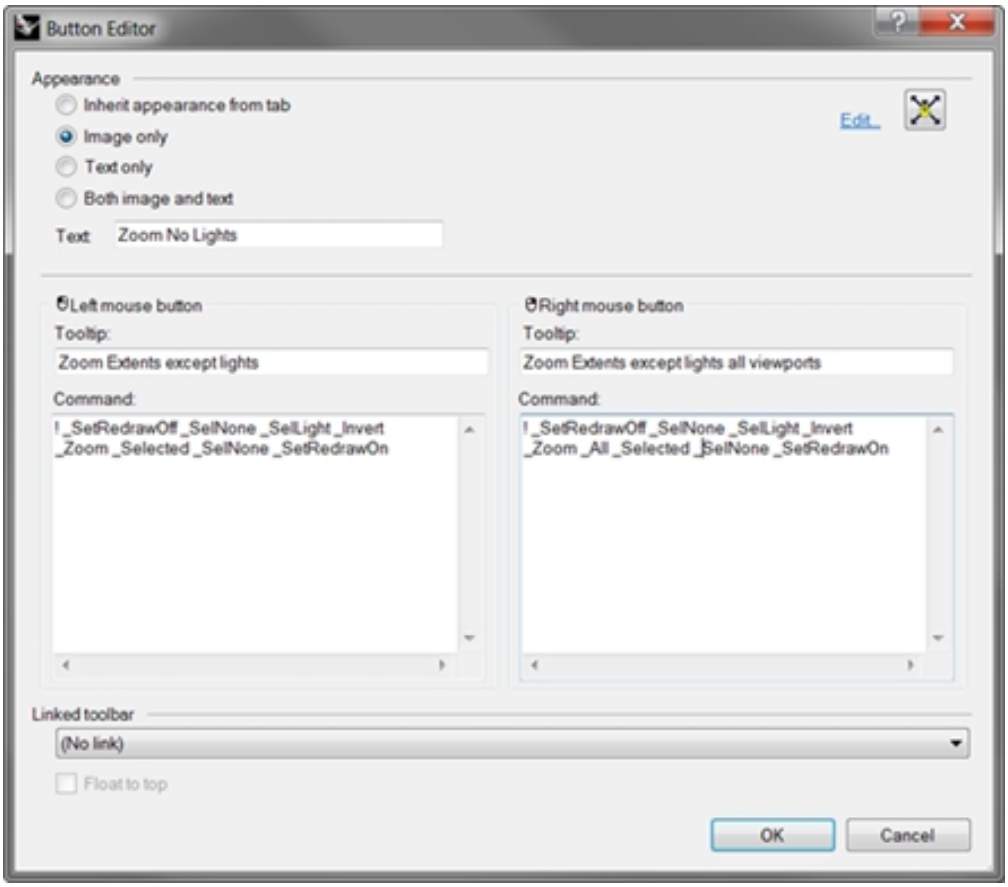


Debug the new button

Combine above macro with **SetRedrawOff**/**SetRedrawOn**. This will make the macro run without flashing and without too much info being reported to command history.

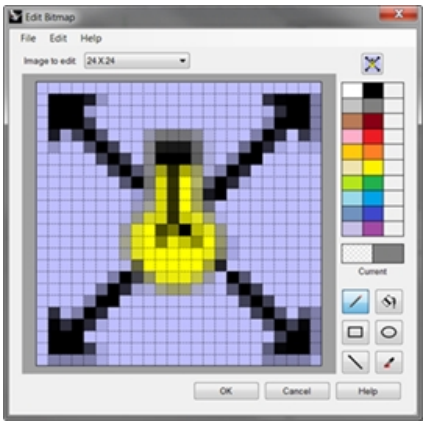
Your macro will work regardless of adding **SetRedrawOff**/**SetRedrawOn**. However, this is good practice to make your macro work more elegantly.

1. In the **Left mouse button Command** box, type
   ! _SetRedrawOff _SelNone _SelLight _Invert _Zoom _Selected _SelNone _SetRedrawOn.
2. In the **Right mouse button Command** box, type
   ! _SetRedrawOff _SelNone _SelLight _Invert _Zoom _All _Selected _SelNone _SetRedrawOn.

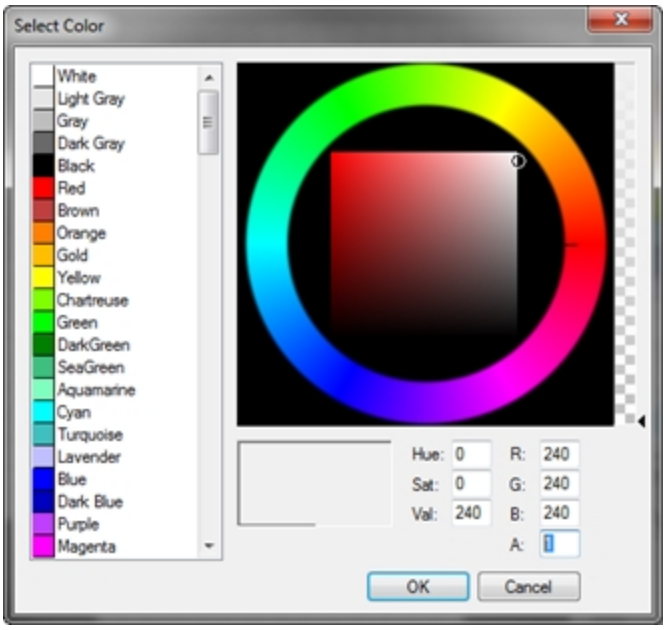


Change the button bitmap image

1. In the **Button Editor** dialog box, click **Edit** next to the button icon in the upper right to open the **Bitmap Editor**. The bitmap editor is a simple paint program that allows editing of the icon bitmap. It includes a Grab function for capturing icon-sized pieces of the screen, and an import file function.
2. In the **Edit Bitmap** dialog box, from the**File** menu, click **Import Bitmap to Fit**, and select the **ZoomNoLights_32.bmp**.
   You can import any bitmap image. If the bitmap is too large, it will be scaled to fit as it is imported.



3. In the **Edit Bitmap** dialog box, make any changes to the picture, and click OK.
   Double-click on the color swatches to the right of the standard color bar to access the **Select Color** dialog box for more color choices.

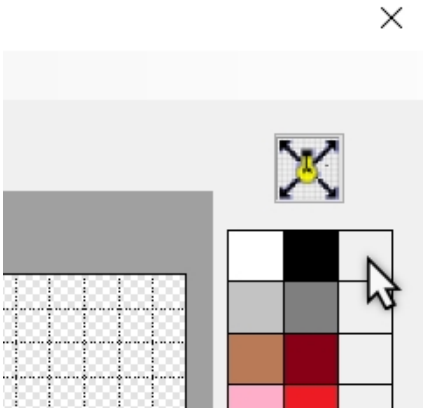


4. In the **Select Color** dialog box, click OK.

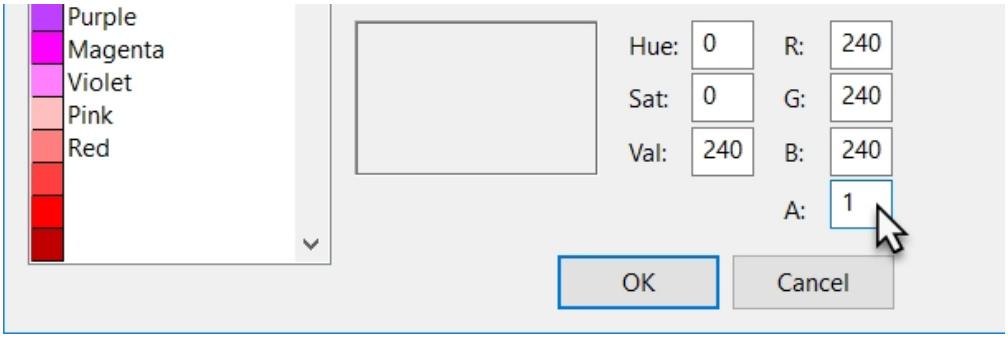
Change the bitmap image to use an alpha channel

Notice that the new button's background color does not match the background color of the other buttons. We will change the image background using an alpha channel, so that it matches the Windows **3D Objects color** like the other buttons.

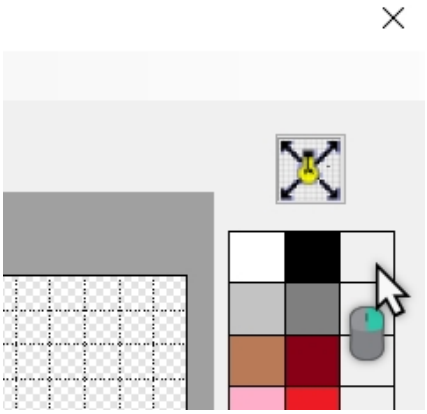
1. Hold down the Shift key and right-click the **ZoomNoLights** button.
2. In the **Button Editor** dialog box, click **Edit** to open the **Bitmap Editor**.
3. Double-click the upper right color swatch to the right of the black color.



4. Change the alpha color number, labeled **A**, for the button color from 255 to 1.
   This will make the current paint color transparent.



5. Right-click over the new alpha color to set the alpha color to the right mouse click.



6. Change to the **Fill** tool, and then right-click in the background area of the button image.

7. In the **Edit Bitmap** dialog box, click OK.
8. In the **Button Editor** dialog box, click OK.
   The color matches the Windows 3D Objects color.

Use the new button

1. Click the **ZoomNoLights** button.
2. Use the button to zoom the model two ways.
   Notice that Zoom Extents ignores the lights.

Rules for commands in buttons

You can enter the commands or command combinations in the appropriate boxes, using these rules:

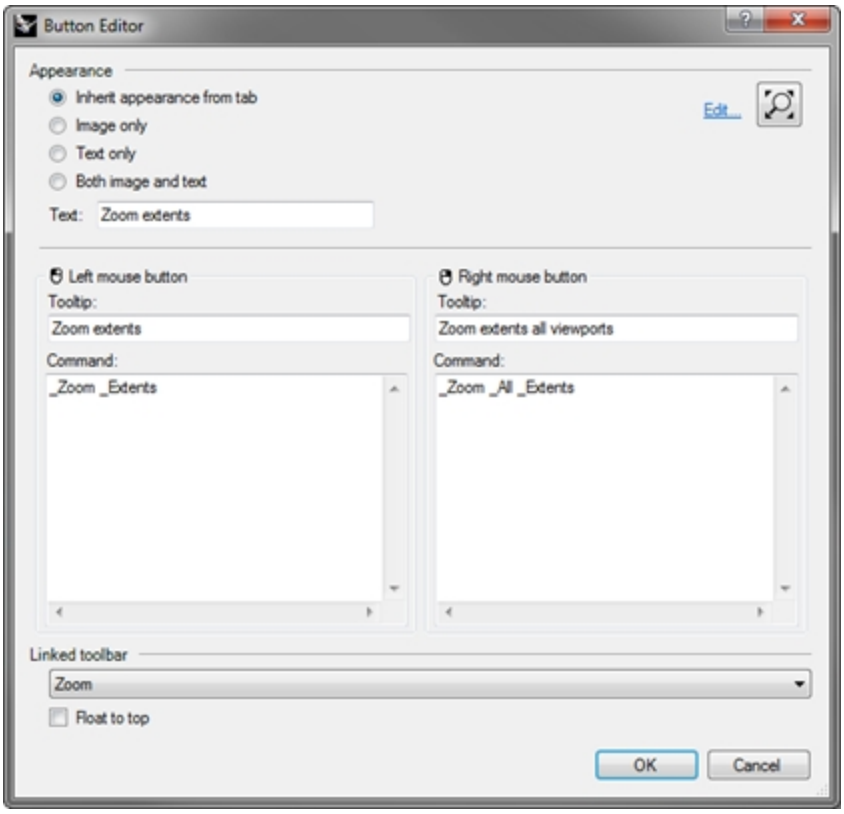
Item	Sample	Description
Space	!_Line	A space is interpreted as Enter. Commands do not have spaces, but you must leave a space between commands.
" "		If your command string refers to a file, toolbar, layer, object name, or directory for which the path includes spaces, the path, toolbar name, or directory location must be enclosed in double-quotes.
! (Exclamation mark)	!-_circle	An ! (exclamation mark) followed by a space is interpreted as Cancel. Generally, it is best to begin a button command with an exclamation mark (!) if you want to cancel any other command that may be running when you click the button.
' (apostrophe)		View manipulation commands like Zoom can be run in the middle of other commands. For example, you can zoom and pan while picking curves for a loft. An '(apostrophe) prior to the command name indicates that the next command is a nest able command.
_ (underscore)		An underscore (_) runs a command as an English command name. Rhino can be localized in many languages. The non-English versions will have commands, prompts, command options, dialog boxes, menus, etc., translated into their respective languages. English commands will not work in these versions. For macros written in English to work on all computers (regardless of the language of Rhino), the macros need to force Rhino to interpret all commands as English command names, by using the underscore.
- (Hyphen)	-_Sweep2	Commands with dialog boxes can be run at the command-line with command-line options. To suppress the dialog box and use command-line options, prefix the command name with a hyphen (-).
Pause		User input and screen picks are allowed in a macro by putting the **Pause** command in the macro. Commands that use dialog boxes, such as Revolve, do not accept input to the dialog boxes from macros. Use the hyphen form of the command (-Revolve) to suppress the dialog box and control it entirely from a macro.

Note: These rules also apply to scripts run using the **ReadCommandFile** command and pasting text at the command prompt. More sophisticated scripting is possible with the Rhino Script plug-in, but quite a lot can be done with the basic commands and macro rules.

Some very useful commands for macros are **SelLast**, **SelPrev**, **SelName**, **Group**, **SetGroupName**, **SelGroup**, **Invert**, **SelAll**, **SelNone**, **ReadCommandFile**, and **SetWorkingDirectory**.

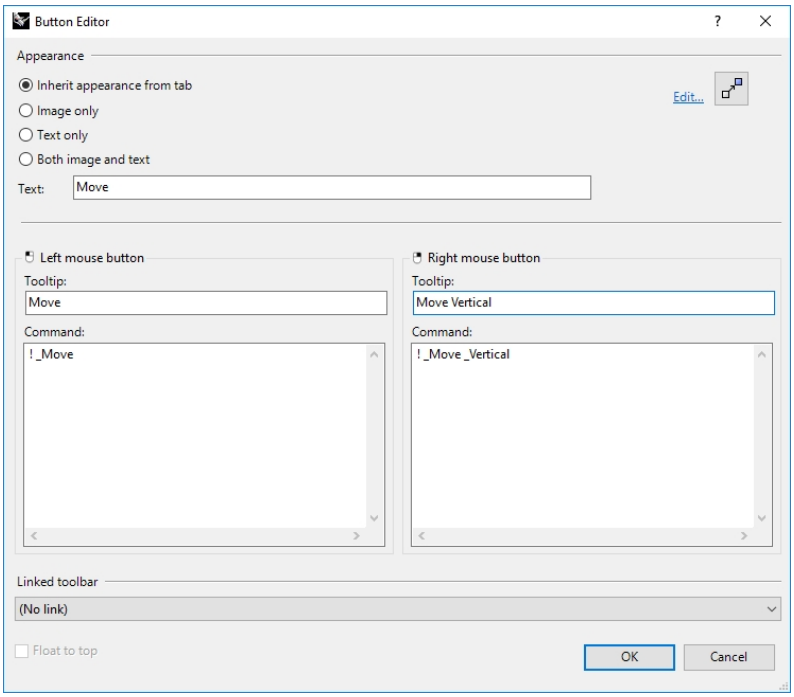
Link a toolbar to a button

1. Hold the Shift key + **Right-click** the **Zoom Extents** button in the **Standard** toolbar.
2. In the **Button Editor** dialog box, click in the **Linked toolbar** area, select **Zoom** from the list, and click OK.
   Now the **Zoom Extents** button has a small black triangle in the lower right corner indicating it has a linked toolbar.
3. Click and hold the **Zoom Extents** button to fly out your newly-created single button toolbar.
   If you close the **Zoom** toolbar you just created, you can always re-open it using the linked button.
4. Try the new linked button.



Add a command to an existing button

1. Hold the Shift key, and on the **Main** toolbar, right-click the **Move** button .
2. In the **Button Editor** dialog box, in the **Right mouse button Command** edit box, type **! _Move _Vertical**

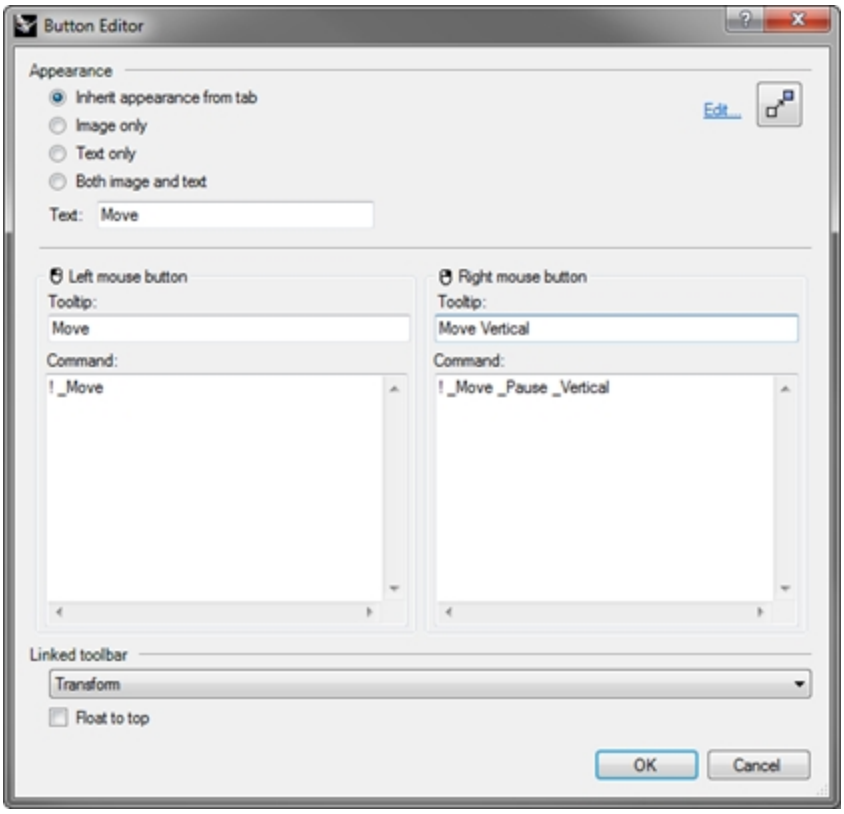


3. In the **Button Editor** dialog box, in the **Right mouse button Tooltip** edit box, type **Move Vertical**.
   This button lets you move objects vertical to the current construction plane. We will use this command several times during the class.
4. Select one of the objects in the model and right-click the **Move** button.
5. Move the selected object vertically to the construction plane.

6. Try it again. However, do not preselect an object in your rmodel.
7. Right-click the **Move** button.
8. The command echos this in the command-line:

```
Command: _Move
Select objects to move: _Vertical
Unknown command: _Vertical
Select objects to move:
```

   There are no objects selected, but the macro inputs "vertical" at the "select objects" prompt. That will not work. You need to pause the command so you can pick the objects.
9. In the **Button Editor** dialog box, in the **Right mouse button Command** edit box, update the macro to include a pause after the **Move** command and before the vertical option
   ! _Move _Pause _Vertical
10. Test with both pre-select and object selection in the command.



Add a new button to the Properties toolbar

Next, you will make a new command button in the **Properties** toolbar.

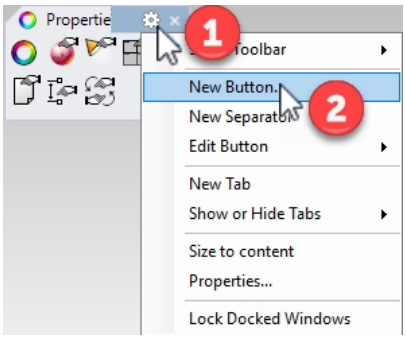
- The left mouse button will change the color property of an object with the Color Picker.
- The right mouse button will change the color property with the RGB value.

Prepare for writing these macros by going through the commands in Rhino that will be required to change the color of an object with the scriptable version of the **Properties** command, **-Properties**.

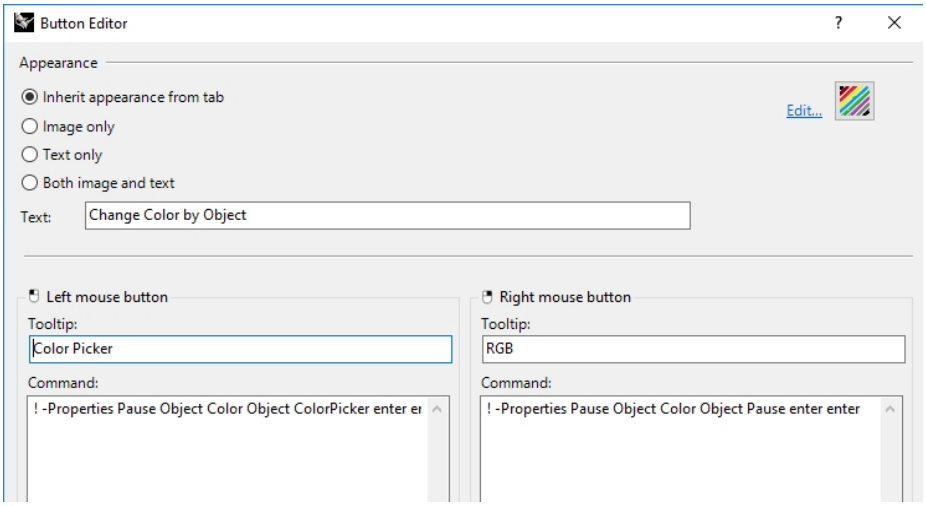
Take notes in a text file and prepare a list of options that are required for the macros.

1. In **Options**, off the **Toolbar** page, select the Default RUI file, and under the Toolbar area scroll to Properties. Click the check box to open the **Properties** toolbar.
2. Click the Ok button on the **Options** dialog. The **Properties** toolbar will appear.

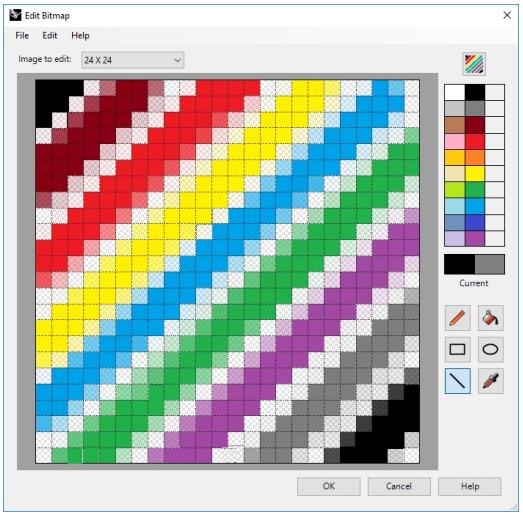
3. On the title bar of the**Properties** toolbar, pick the gear and from the menu click **New Button**.



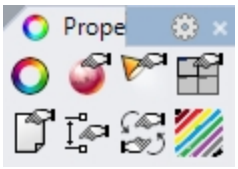
4. In the **Button Editor** dialog, in the Text field, type **Change Color by Object.**
5. In the **Left mouse button Tooltip**, type **Color Picker** and in the **Right mouse button Tooltip**, type **RGB**
6. In the **Left mouse button Command**, type:
   ! -_Properties _Pause _Object _Color _Object _ColorPicker _Enter _Enter
7. In the **Right mouse button Command**, type:
   ! -_Properties _Pause _Object _Color _Object _Pause _Enter _Enter
   Note: You will find these macros in a Level 2 materials folder in text file **Macros.txt**.



8. In the upper right corner of the **Button Editor** dialog, click **Edit**.
9. In the **Edit Bitmap** dialog box, use the tools to design a button that would be representative of these macros. Here is one suggestion:



10. Click Ok to close the **Edit Bitmap** dialog and your button will update.



11. Test the button macros.

Command aliases

The same commands and macros that are available for buttons are also available for command aliases. Command aliases are like using shorthand in Rhino. They are commands and macros that are activated whenever commands are allowed, but are often used as a keyboard shortcut.

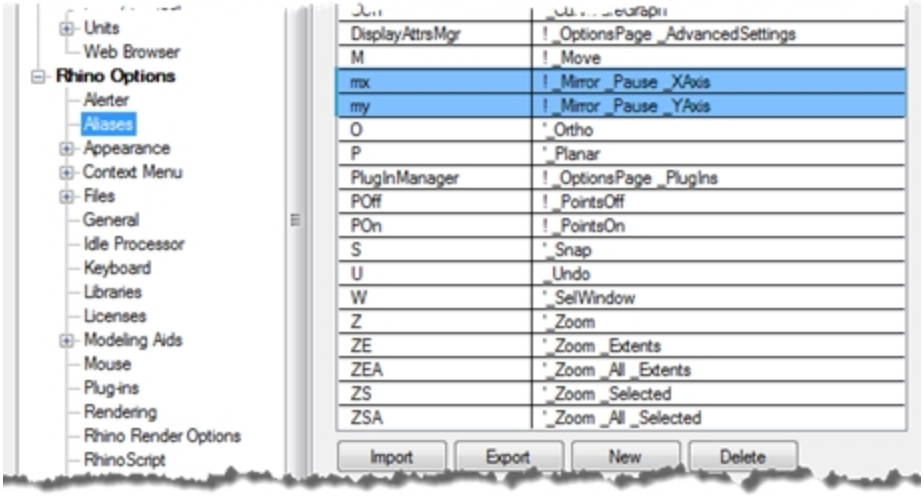
Use aliases for command sequences that you use often or frequently.

Note: When making aliases, use keys that are close to each other or repeat the same character 2 or 3 times, so they will be easy to use.

For the next few sections in this chapter, you will find a text file in the folder containing your Level 2 materials called **Macros.txt**. If you need to trouble shoot your macros, you can open it and compare the content to your entries.

Make a command alias

1. **Open** the model **Aliases.3dm**.
2. On the **Tools** menu, click Options.
3. In the **Rhino Options** dialog box, on the **Aliases** page, you can add aliases and command strings or macros.
4. Click **New** to make a new alias.
   We will make aliases to mirror selected objects vertically and horizontally across the origin of the active construction plane. These are handy when making symmetrical objects built centered on the origin.



5. In the **Alias** column, type **mx**.
6. In the **Command Macro** column, type **! _Mirror _Pause _XAxis**
   The alias is in the left column and the command string or macro is in the right column. The same rules apply here as with the buttons.
   Aliases can be used within other aliases' macros or button macros.
7. To make another new alias, click the New button.
8. In the **Alias** column, type **my**.
9. In the **Command Macro** column, type
   ! _Mirror _Pause _YAxis

Try the new aliases

- Select some geometry, and type **mx** or **my**, and press Enter.
  If no objects are pre-selected, the Pause in the script prompts you to select objects, and press Enter to complete the selection set.

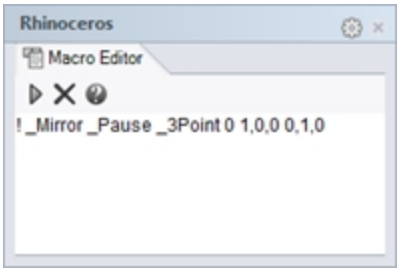
Macro editor

When making macros that are more complicated, it is good practice to use Rhino's built-in macro editor. Macros can be edited and run directly from the editor. This allows you to quickly test whether command options and syntax are correct.

In the following example we will make a mirror macro that allows you to mirror across the construction plane. We will use the **Macro Editor** to build and test the macro before we add it to the Alias list.

Use the macro editor

1. On the **Tools** menu, click **Command**, and then click **Macro Editor**.
2. In the **Macro Editor** type
   ! _Mirror _Pause _3Point 0 1,0,0 0,1,0.



3. To test the macro, click the **Run** icon in the **Macro Editor**. 
4. If the macro runs as expected, select the text, and copy it to the clipboard.
5. Open the **Options** dialog box, and on the **Alias** page, and make a new alias **mc**.
6. Paste the text from the **Macro Editor** into the **Alias** command column.
7. Select some geometry, and test the new alias.
8. Type **mc**, and press Enter.

Export and import Rhino options

There are times when you might want to copy all or some of the options from one computer to another. An example might be a desktop computer and a laptop computer. This is especially true for aliases, keyboard shortcuts, and display modes. Rhino has commands that Export options to a file as well as Import options from a file.

Export options

1. On the **Tools** menu, click **Export Options**.
2. In the **Save As** dialog box, for the **File Name,** type **Level2_Options**.
   The current options are saved to a file.

Import options

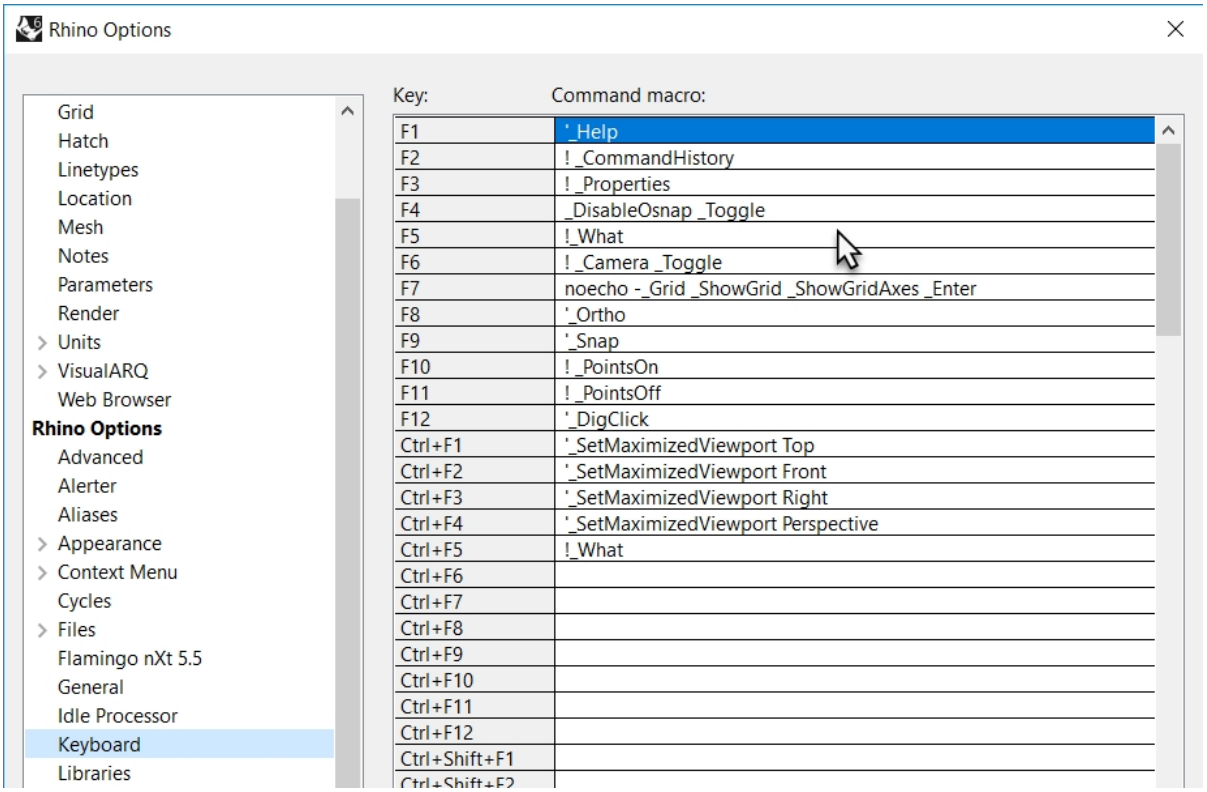
1. **Delete** one of the aliases you previously made.
2. On the **Tools** menu, click **Import Options**.
3. In the **Import Options** dialog box, select the file you just saved.
4. For the **Options to import**, click **Aliases**, **Appearance**, or any other options you wish to import.
5. Check to see if the alias you deleted is back.

Shortcut keys

The same commands, command strings, and macros that you can use for buttons and aliases are also available for keyboard shortcuts. Shortcuts are commands and macros that are activated by certain combinations of function keys, **Ctrl**, **Alt**, and alphanumeric keys.

Make a shortcut key

1. On the **Tools** menu, click **Options**.
2. In the **Rhino Options** dialog box, on the **Keyboard** page, you can add command strings or macros.
3. To make a new shortcut, click the **Command macro** column next to **F4**.
4. For the shortcut, type **_DisableOsnap _Toggle**.
   This shortcut will make it easy to toggle the state of running object snaps.
5. To make a new shortcut for the **What** command, click the **Command macro** column next to **F5**.



6. For the shortcut, type **!_What** command.
7. Pick **Ok** to close the the dialog box. Test the **F4** and **F5** keys.
   Preselect an object to test the **What** command on **F5**.
   There are several shortcut keys that already have commands assigned. The same rules apply here as with the buttons and aliases.

Plug-ins

Plug-ins are programs that extend the functionality of Rhino.

Plug-in classifications include:

Included plug-ins

Shipped and installed with Rhino. Some of these plug-ins are loaded, for example Rhino Render, Render Development Kit, Rhino Toolbars and Menus, BoxEdit, etc. Others are installed, but not loaded. Most of these plug-ins are Import/Export plug-ins. They are generally enabled and will get loaded when they are used for the first time.

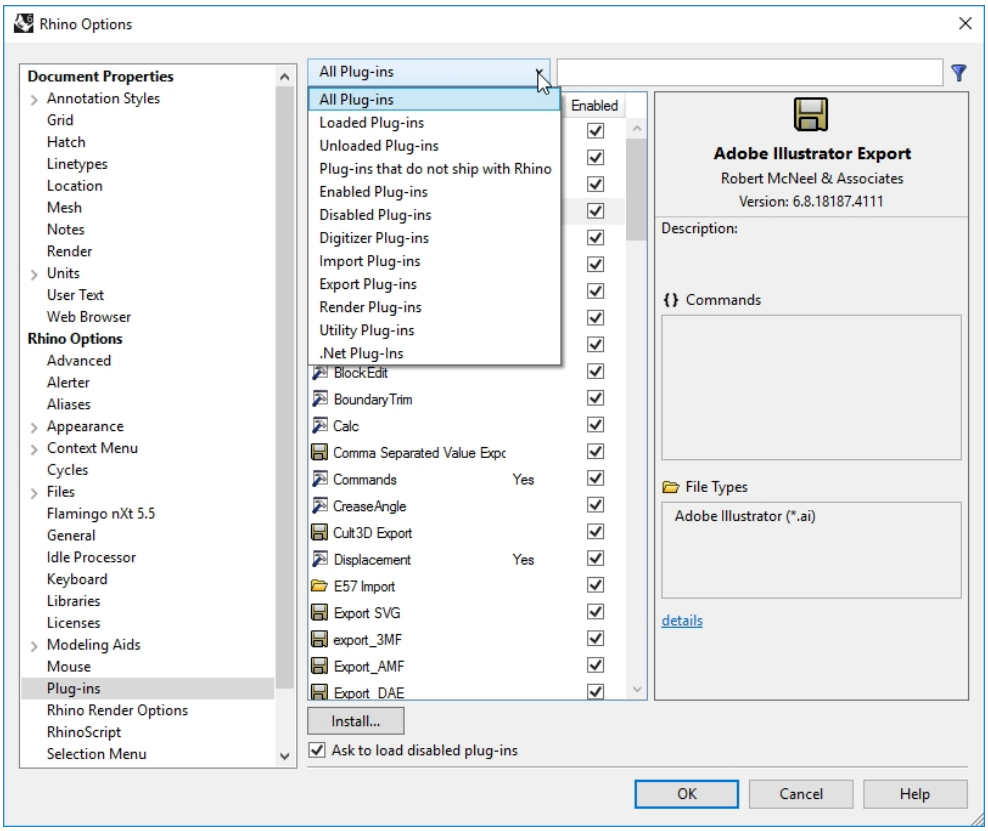
McNeel plug-ins

Flamingo nXt, Penguin, Brazil (rendering) and Bongo (animation) are McNeel products that are available for purchase.

Third-party plug-ins

These are programs and utilities that are developed by third-party developers. Some of these are free, but most are available for purchase. A few of the programs are stand-alone applications that work with Rhino, but are not plug-ins.

Generally, they add some specific capability to Rhino. For example RhinoCam is a CAM application, VRay is a rendering application, RhinoGold is jewelry design software, VisualARQ is for building architectural models, etc. All of these are developed by experts in their specific industries. For more information about these programs visit the Food4Rhino website.



Load a plug-in

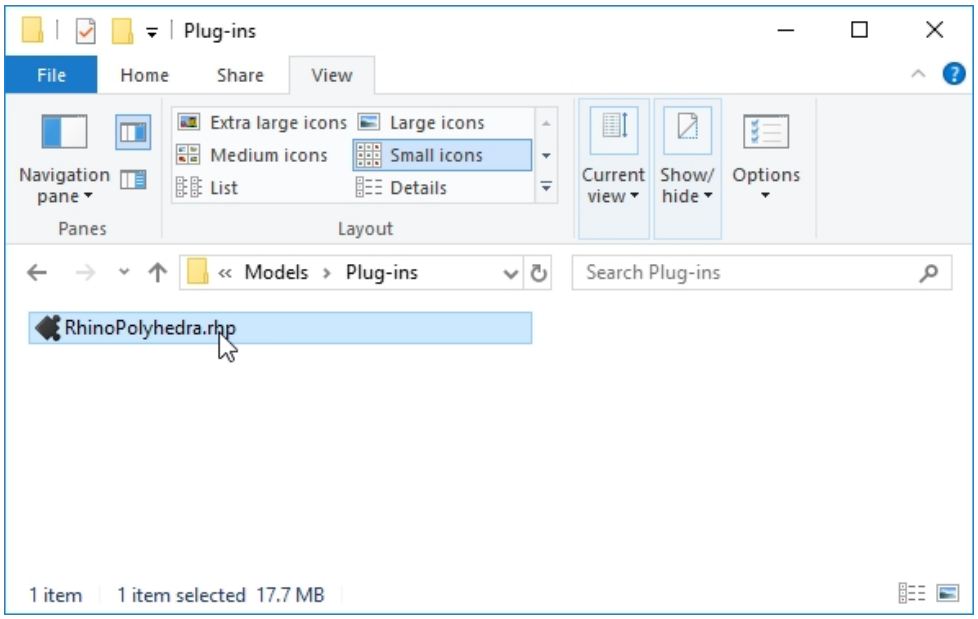
For this example we have included a plug-in from the Rhino 5.0 labs page for you to install and use.

1. On the **Tools** menu, click **Options**.
2. Click **Plug-ins**.
   A list of currently loaded and available plug-ins is displayed.
3. On the **Plug-ins** page, click **Install**.
4. In the **Load Plug-In** dialog box, navigate to the **Level 2/Models/Plug-ins** folder, then **rhinopolyhedra.rhp**.
   Note : With an account on Food4Rhino, you may also download the **Rhinopolyhedra.rhi** plug-in. The rhi is a Rhino installer file that will need to be double-clicked and installed outside of Rhino.

Load a plug-in using drag and drop

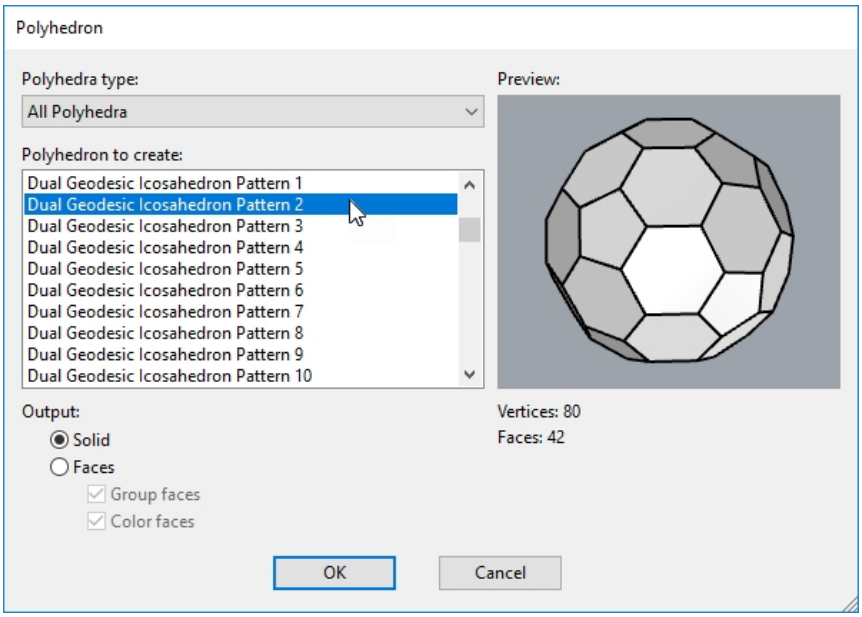
For the initial load, you may also drag and drop the **rhinopolyhedra.rhp** from File Explorer. This will only work if you have not registered **Rhinopolyhedra** in a previous session.

1. Open a **Windows Explorer** window.
2. Navigate to the **Level 2/Models/Plug-ins** folder or any folder that has a **.rhp** plug-in that you want to install.
3. Click and hold on the plug-in file **rhinopolyhedra.rhp**, then drag it and drop it into the open Rhino application window.

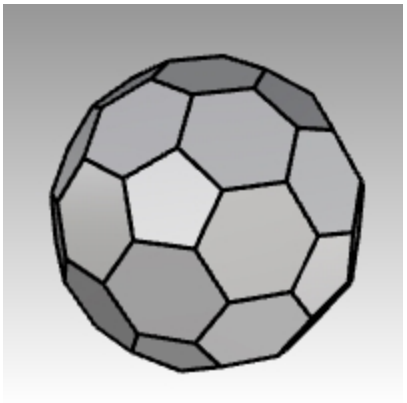

Plug-ins
File
Home
Share
View
Navigation pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Layout
Current view
Show/ hide
Options
« Models › Plug-ins
Search Plug-ins
RhinoPolyhedra.rhp
1 item
1 item selected 17.7 MB

Run the command

1. To run the command in Rhino, type **Polyhedron** on the command-line.
2. In the **Polyhedron** dialog, select one of the polyhedrons from the list, such as **Dual Geodesic Icosahedron Pattern 2**.



3. Click a center point and a radius point to complete the polyhedron.



Scripting

Rhinoceros supports scripting using **RhinoScript** and **Rhino.Python**.

To script Rhino, you must have some programming skills. Fortunately, scripting is easy to learn and there are materials available to help you get started.

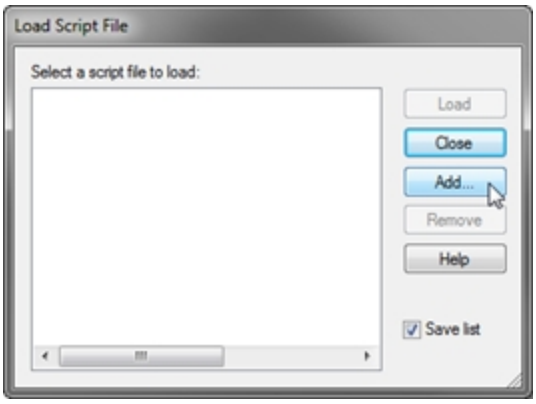
You will find more details on the Developer Wiki.
In addition, Rhino installs with Help for both scripting tools. See the **EditScript** and **EditPythonScript** commands for details.

We will not cover how to write a script in this class, but we will learn how to run a script and apply it to a button.

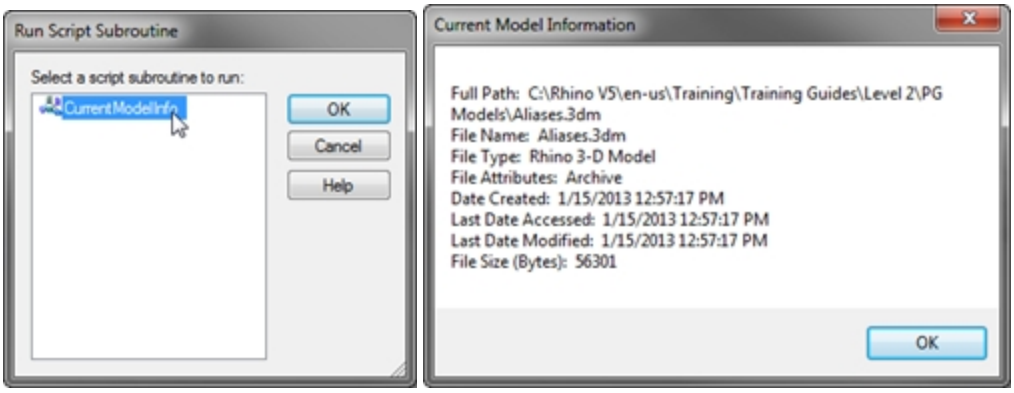
The following script will list information about the current model.

Load a RhinoScript

1. On the **Tools** menu, click **RhinoScript**, then click **Load**.
2. In the **Load Script File** dialog box, click **Add**.
3. In the **Open** dialog box, select **CurrentModelInfo.rvb**, then click Open.
   Note: You may get the following message, "Cannot find the script file CurrentModelInfo.rvb."
   If that happens you will need to include the full path to the folder where the script file is located or add a search path in the **Files** section of **Rhino Options**.
4. In the **Load Script File** dialog box, highlight **CurrentModelInfo.rvb**, then click Load.



5. **Save** the current model.
   If you do not have a saved version of the model, no information is possible.
6. On the **Tools** menu, click **RhinoScript**, then click **Run**.
7. In the **Run Script Subroutine** dialog box, click **CurrentModelInfo** and then click OK.
   A dialog box describing the current information about this model displays.

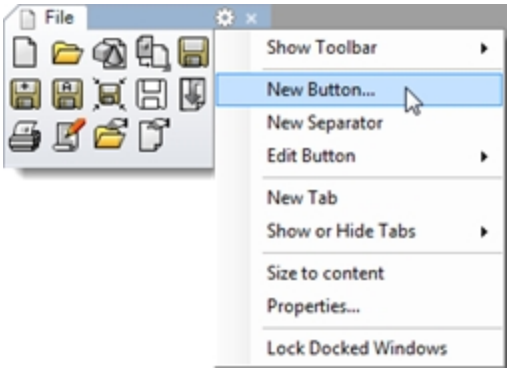


Edit the script file

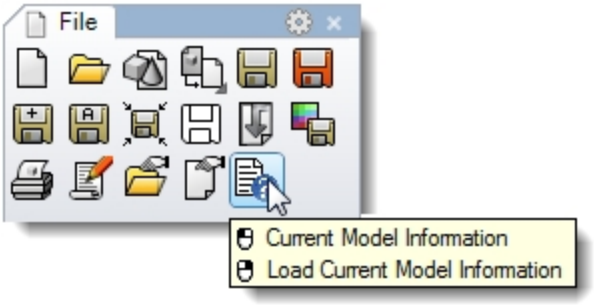
1. On the **Tools** menu, click **RhinoScript**, then click **Edit**.
2. In the **Rhino Script Editor** window, from the **File** menu, click Open.
3. In the **Open** dialog box, select **CurrentModelInfo.rvb**, and click Open.
   We will not be editing script files in this class. This exercise is to show how to access the editing feature if needed.
4. Close the **Rhino Script Editor** window.

Make a button that will load or run a script

1. On the **Tools** menu, click **Toolbar Layout**.
2. In the **Toolbars** dialog box, check the **File** toolbar, and then close the dialog box.
3. Right-click the title bar of the **File** toolbar.
4. On the pop-up menu that appears, click **New Button** command.
5. In the **Button Editor** dialog box, in the **Left mouse button Tooltip** box, type **Current model information**.
6. In the **Right mouse button Tooltip** box, type **Load current model information**.



7. In the **Text** box, type **Model Info**.
8. In the **Left mouse button Command** box, type
   ! -_RunScript (CurrentModelInfo)
9. In the **Right mouse button Command** box, type
   ! -_LoadScript "CurrentModelInfo.rvb"



Add a custom bitmap

1. In the **Button Editor** dialog box, click **Edit**.
2. In the **Edit Bitmap** dialog box, from the **File** menu, click **Import Bitmap**.
3. Choose **CurrentModelInfo.bmp** file, and click Open.
4. In the **Button Editor** dialog box, click OK.
5. Try the new button.

Template files

A template is a Rhino model file you can use to store basic settings. Templates include all the information that is stored in a Rhino 3DM file: objects, blocks, layouts, grid settings, viewport layout, layers, units, tolerances, render settings, dimension settings, notes, and any setting in document properties.

You can use the default templates that are installed with Rhino or save your own templates to base future models on. You will likely want to have templates with specific characteristics needed for particular types of model building.

The standard templates that come with Rhino have different viewport layouts or unit settings, but no geometry, and default settings for everything else. Different projects may require other settings to be changed. You can have templates with different settings for anything that can be saved in a model file, including render mesh, angle tolerance, named layers, lights, and standard pre-built geometry and notes.

If you include notes in your template, they will show in the **Open Template File** dialog box.

The **New** command begins a new model with a template (optional). It will use the default template unless you change it to one of the other templates or to any other Rhino model file.

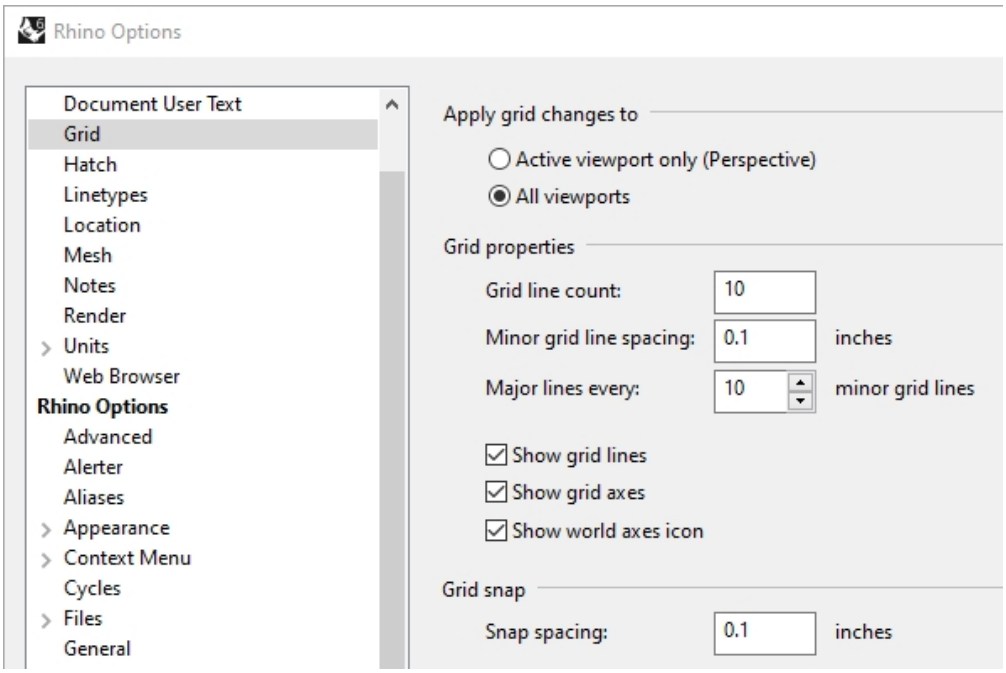
To change the template that opens by default when Rhino starts up, choose New and select the template file you would like to open when Rhino starts, then check the **Use this file when Rhino starts** box.

Exercise 3-2 Create a template

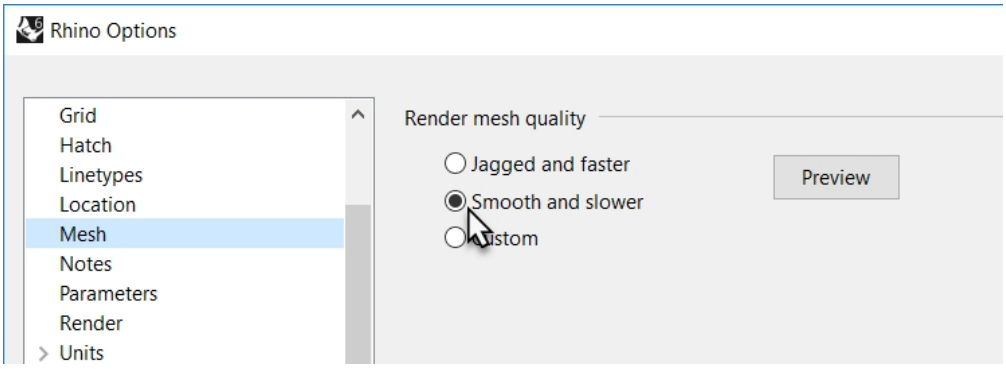
1. Start a new model.
2. As the template, select the **Small Objects - Inches.3dm**.
3. On the **Render** menu, click **Current Renderer**, and then click **Rhino Render**.

Set the document properties

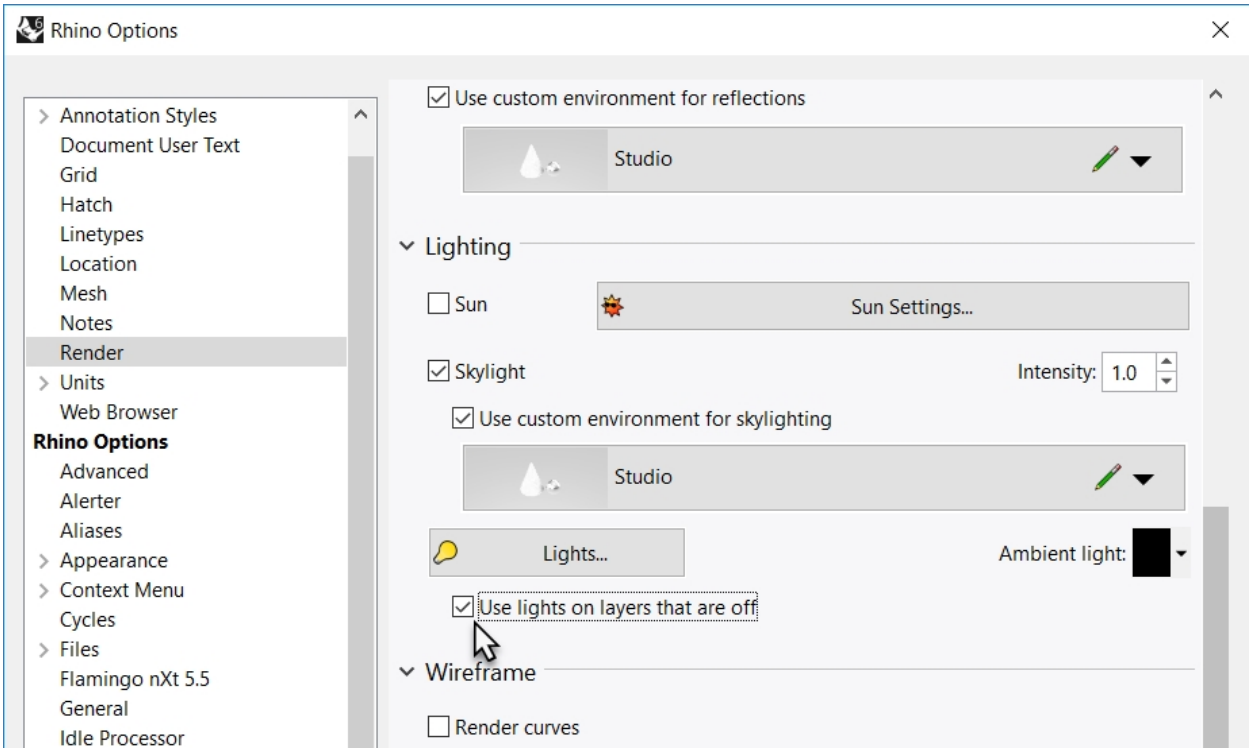
1. On the **File** menu, click **Properties**.
2. In the **Document Properties** dialog box, on the **Grid** page set the following:
   Grid line count to **10**
   Minor grid line spacing to **0.1** inches
   Major lines every to **10** minor grid lines
   Snap spacing to **0.1**.



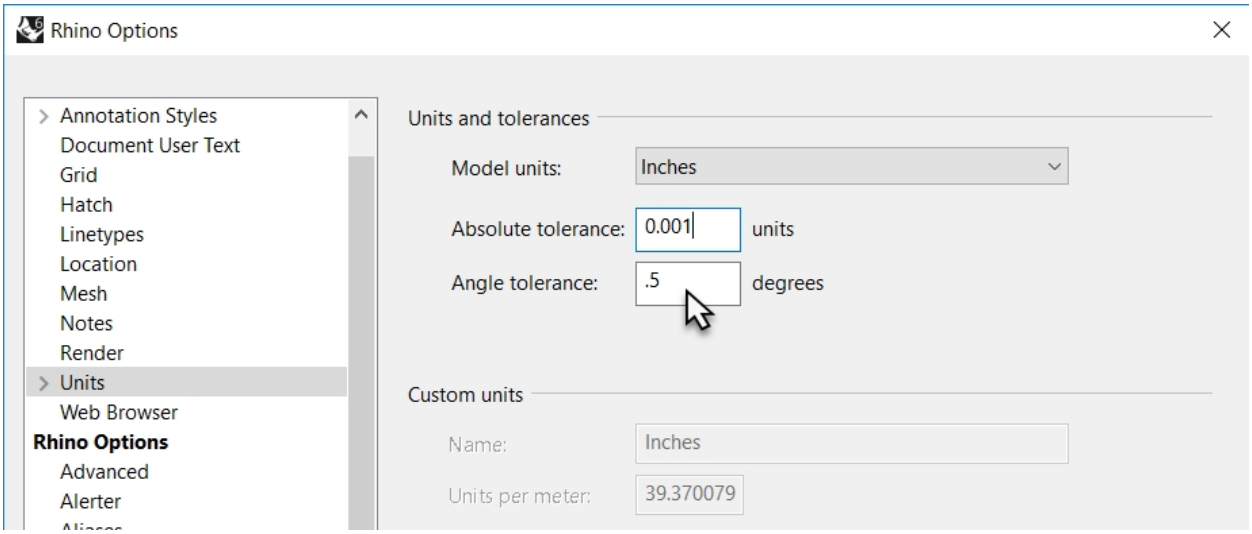
3. On the **Mesh** page change the setting to **Smooth & slower**.



4. On the **Render** page, scroll down to the **Lighting** section. Check **Use lights on layers that are off**.

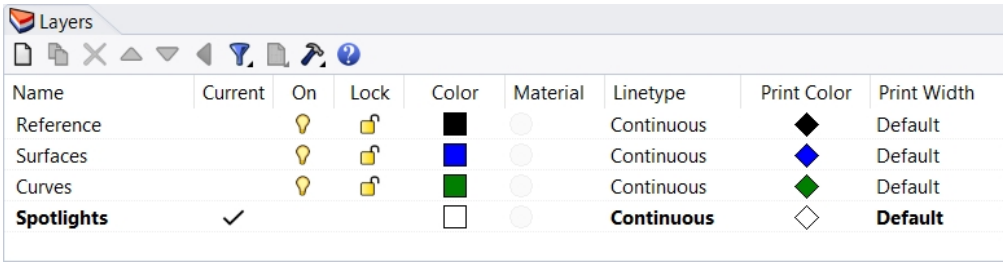


5. On the **Units** page, change the **Angle tolerance** to **0.5**, click OK.
   The end tangent normals will be determined by this setting.

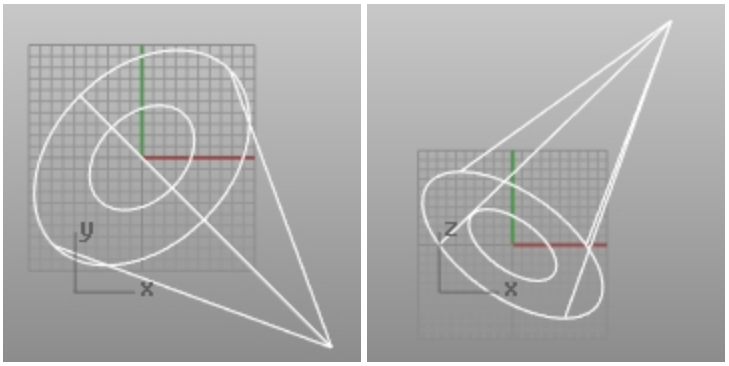


Set up the layers

1. Open the **Layers** panel. Make the following changes:
   Rename **Layer 05** to **Spotlights**
   Rename **Layer 04** to **Curves**
   Rename **Layer 03** to **Surfaces**
   Rename **Default** to **Reference**.
2. Make the **Spotlights** layer current.
3. Delete the unused layers: **Layer 01** and **Layer 02**.



4. Set up a spotlight so that it points at the origin and is approximately 45 degrees in the **Top** viewport and tilted 45 degrees in the **Front** viewport.
5. Use the **my** alias to mirror the light to make a second one.
6. To make the **Curves** layer the only visible layer, from the **Edit** menu, click **Layers** then click **One Layer On**.
7. Select the **Curves** layer.

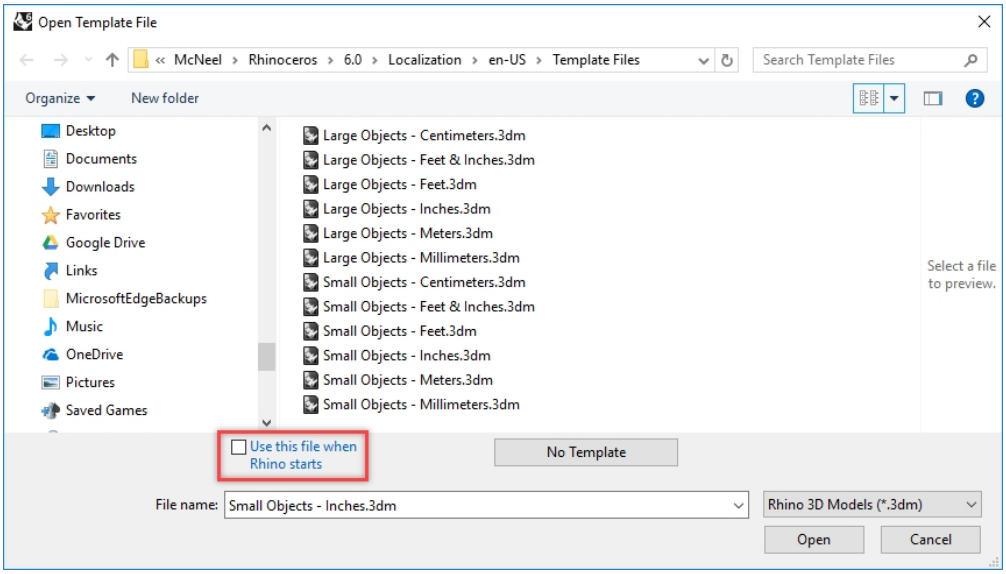


Save notes

1. On the **Panel** menu, click **Notes**.
2. Type the details about this template in the **Notes** panel.
3. On the **File** menu, click **Save As Template**.
4. Name the template **Small Objects – Decimal Inches - 0.001.3dm**.
   This file with all of its settings is now available any time you start a new model.

Set a default template

1. On the **File** menu, click **New**.
2. Select the template you want to use as the default template.
3. In the **Open Template File** dialog box, check the **Use this file when Rhino starts** check box.
   You should make custom templates for the kind of modeling that you do regularly to save set up time.



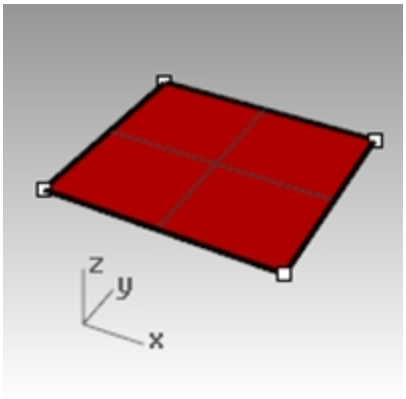
Chapter 4 - NURBS topology

The underlying geometry of NURBS surfaces have a rectangular topology in UV, or parameter space. Rows of surface points, and parameterization, are organized in two directions (U and V). These two directions are crosswise to each other, at more or less 90 degrees on an ideal surface, though this is not always possible. This structure is not always obvious when creating or manipulating a surface. Remembering this structure is useful in deciding which strategies to use when creating or editing geometry.

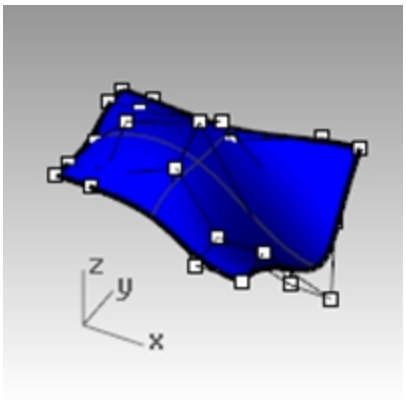
Exercise 4-1 Work with topology

The following exercise demonstrates how NURBS topology is organized and discuss some special cases that need consideration when creating or editing geometry.

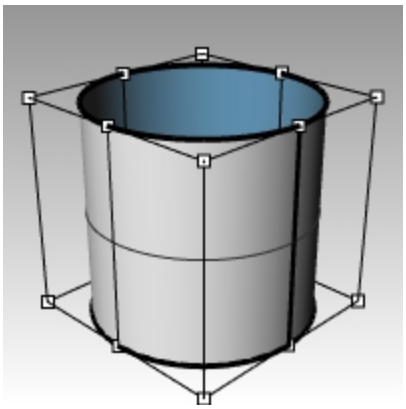
1. **Open** the model **Topology.3dm**.
   Several surfaces are visible on the current layer.
2. Turn on the control points of the simple rectangular plane on the left.
   The surface has four control points, one at each corner—this is a simple untrimmed planar surface that shows the rectangular topology.



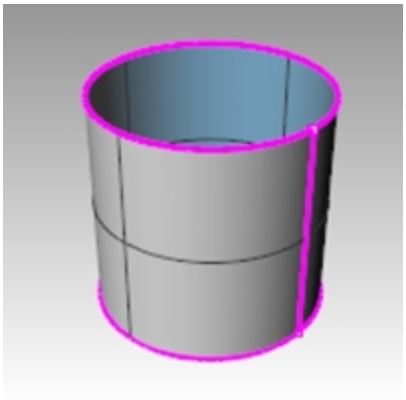
3. Now, turn on the control points of the curvier surface.
   There are many more points, but it is clear that they are arranged in a rectangular fashion.



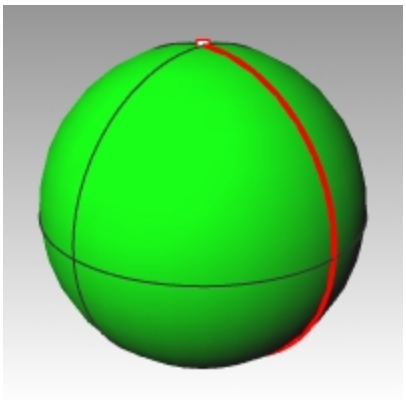
4. Select the **cylinder**.
   It appears as a continuous circular surface, but it also has a rectangular boundary.



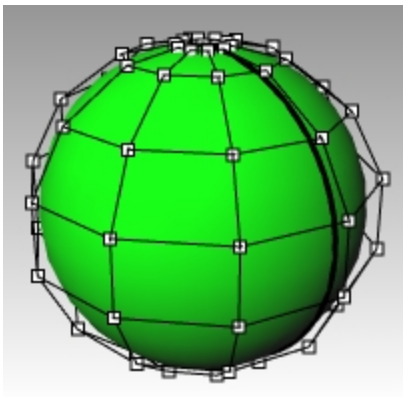
5. Use the **ShowEdges** command *(Analyze menu: Edge Tools > click Show Edges)* to highlight the surface edges. Notice that a seam is highlighted on the cylinder. The seam that is highlighted represents two edges of the rectangle, while the other two edges are circular at the top and the bottom. The rectangular topology is present here, also.



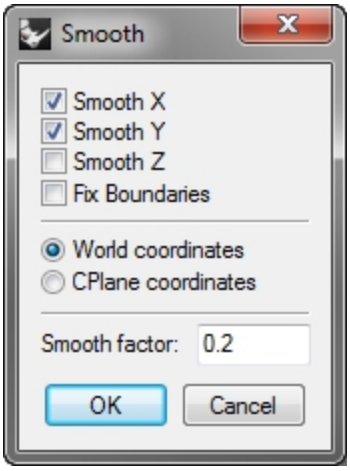
6. Select the sphere.
   It appears as a closed continuous object.
7. Use the **ShowEdges** command to highlight the edges.



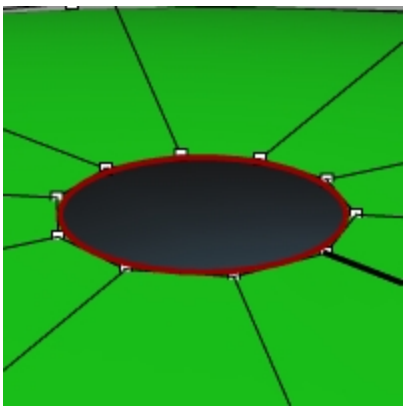
Notice that a seam is highlighted on the sphere. The highlighted seam represents two edges of a rectangular NURBS surface, while the other two edges are collapsed to a single point at the poles. When all of the points of an untrimmed edge are collapsed into a single point, it is called a singularity.
The rectangular topology is present here, also, though very distorted.



8. Turn on the **Control Points** for the sphere.
9. **Zoom Target** *(View menu: Zoom > Zoom Target)* draw a select window very tight around one of the poles of the sphere.
10. Select the point at one pole of the sphere and start the **Smooth** *(Transform menu: Smooth)* command.
11. In the **Smooth** dialog box, clear the **Smooth Z** check box, and click OK.



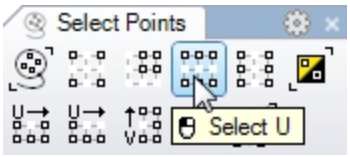
A hole appears at the pole of the sphere. There's no longer a singularity at this pole of the sphere. ShowEdges will highlight this as an edge as well.



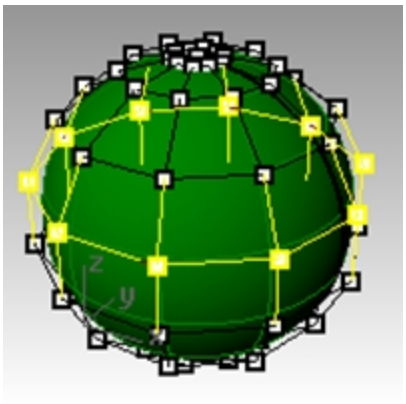
12. Use the Home key to zoom back out.
This is the fastest way to step back through view changes.

Select points

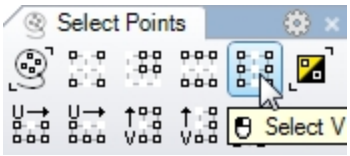
1. Open the **Select Points** toolbar.



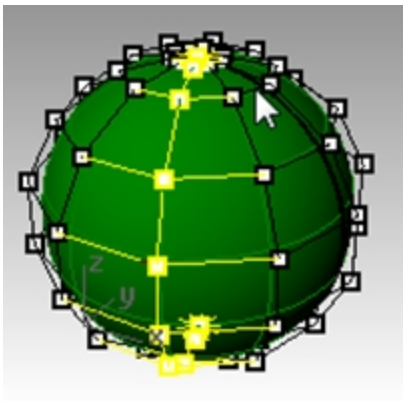
2. Select a single point at random on the sphere.
3. On the **Select Points** toolbar, click **Select U**.
   An entire row of points is selected.



4. Clear the selection by clicking in an empty area and select another point on the sphere.
5. On the **Select Points** toolbar, click **Select V**.

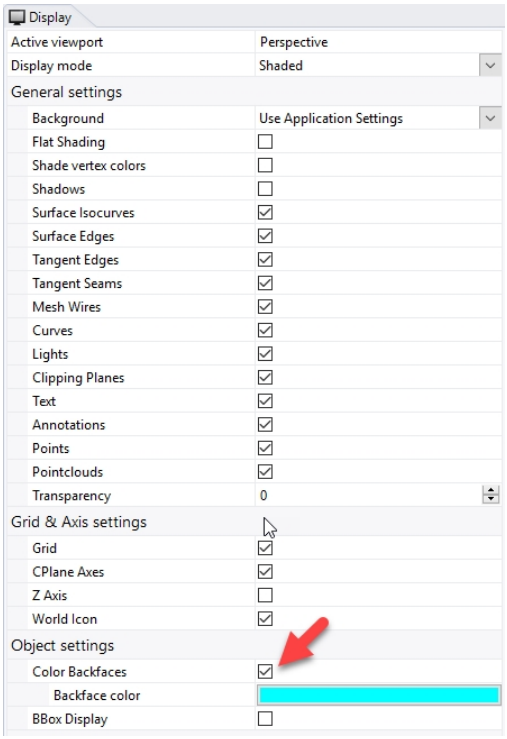


   A row of points in the other direction of the rectangle is selected. This arrangement into u- and v-directions is always the case in NURBS surfaces.
6. Try the other buttons in this toolbar on your own.



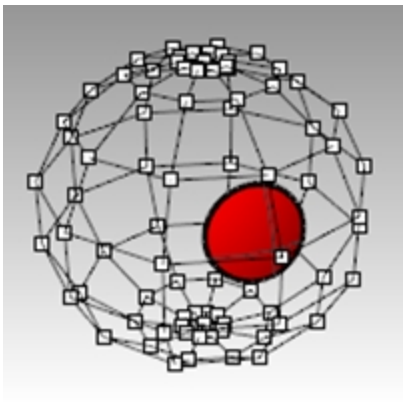
Tip: To see at a glance which way the current surface normals are pointing, set any shaded display mode to show colored backfaces. This is one of the settings available in the Display panel (*Panels menu >Display*) for shaded

modes.
Select a bright, distinguishing color like orange, yellow, or cyan.

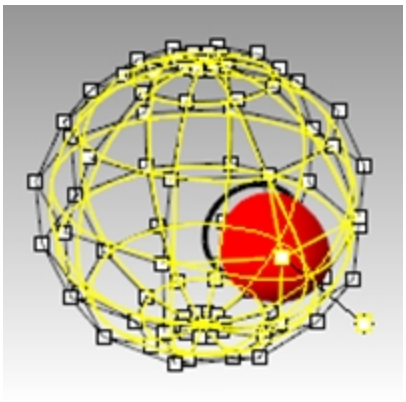


Exercise 4-2 Observe trimmed surfaces

1. Open the model **Trimmed NURBS.3dm**.
   This surface has been trimmed out of a much larger surface. The underlying four sided surface data is still available after a surface has been trimmed, but it is limited by the trim curves (edges) on the surface.



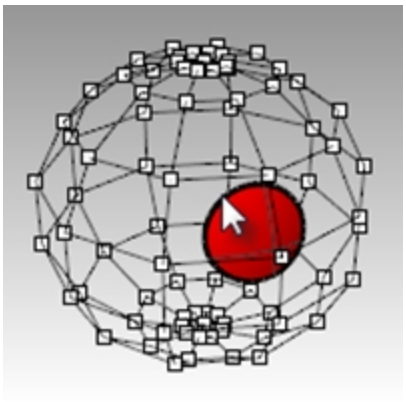
2. Select the surface and turn on the control points, then drag a few control points.
   Control points can be manipulated on the trimmed part of the surface or the rest of the surface, but notice that the trimming edges move around as the underlying surface changes. The trim curve always stays on the surface.



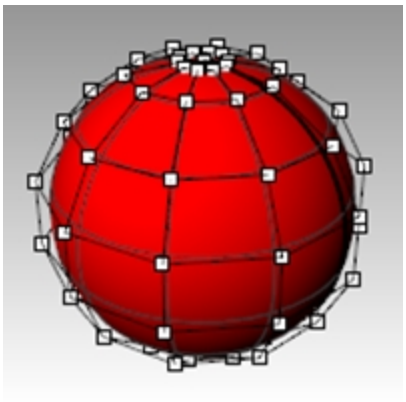
3. Use the **Undo** command to undo the point manipulation.

Remove the trims from a surface

1. Start the **Untrim** *(Surface menu: Surface Edit Tools > Untrim)* command.
2. Select the single edge of the trimmed surface.
   The original underlying surface appears and the trim boundary disappears.

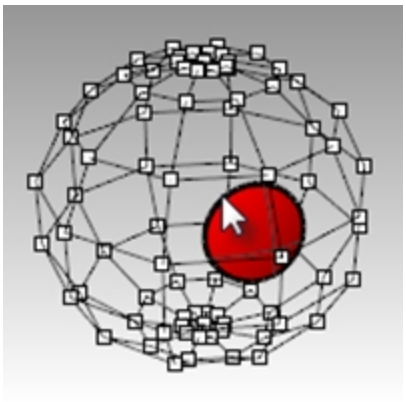


3. Use the **Undo** command to return to the previous trimmed surface.

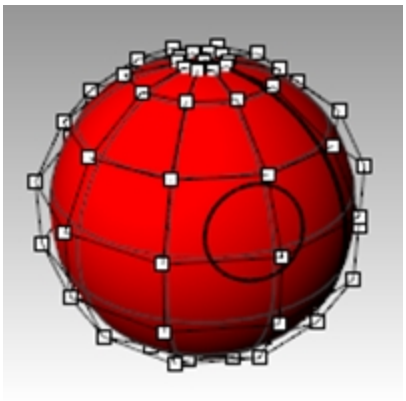


Detach a trimming curve from a surface

1. Start the **Untrim** command with the **KeepTrimObjects** option set to **Yes** *(Surface menu: Surface Edit Tools > Detach Trim)*.
2. Select the edge of the surface.
   The original underlying surface appears. The boundary edges are converted to curves that are no longer associated with the surface.

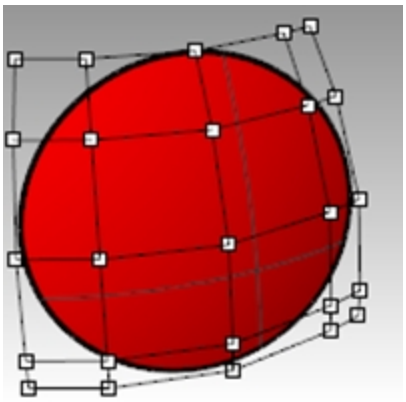


3. **Undo** to return to the previous trimmed surface.



Shrink a trimmed surface

1. Start the **ShrinkTrimmedSrf** command *(Surface menu: Surface Edit Tools > Shrink Trimmed Surface)*.
2. Select the surface, and press Enter to end the command.
   The underlying untrimmed surface is replaced by a one with a smaller range that matches the old surface exactly in that range. You will see no visible change in the trimmed surface. Only the underlying untrimmed surface is altered.



Custom display modes

Assigning a custom color to the backface of a surface in the display mode will help you know the direction of the normal by the display color.

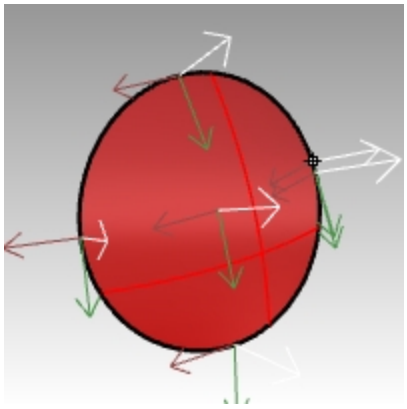
Customizing a display mode will make this preview available from any viewport in any model on your computer that uses your custom display mode.

You will start by customizing the Shaded display mode.

Exercise 4-3 Color the backface

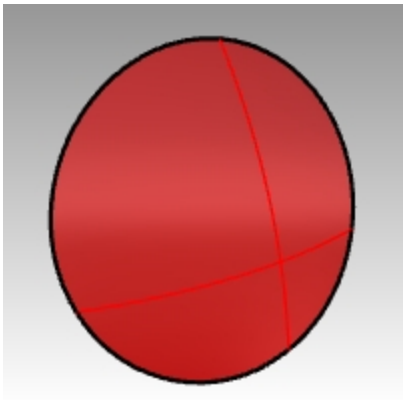
1. If it is not already open, open the model **Trimmed NURBS.3dm**.

2. Right-click on the **Perspective** viewport title, and pick **Shaded**.
3. Select the surface and from the **Analyze** menu, click **Direction**.
4. The surface will be decorated with a 3 sides arrow, similar to construction plane:

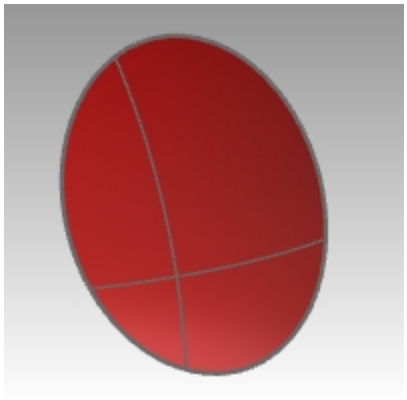


The white arrow shows the normal direction. You can think of the normal as the direction that points "outside" or "up." Enter to exit the command and to return the surface to the normal view.

5. The outside and the inside of the surface are not easy to distinguish in the standard **Shaded** display mode.

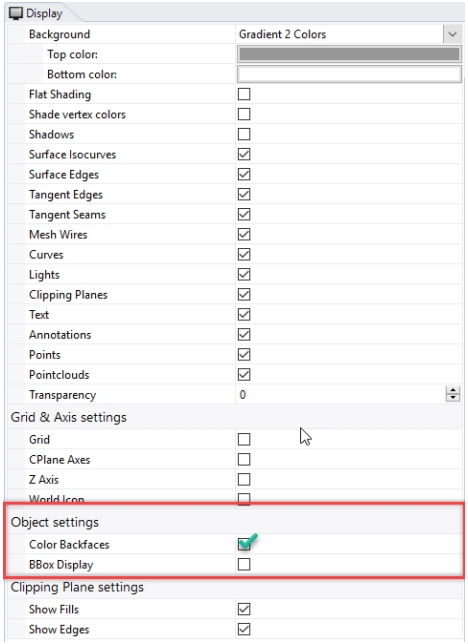


Outside of surface

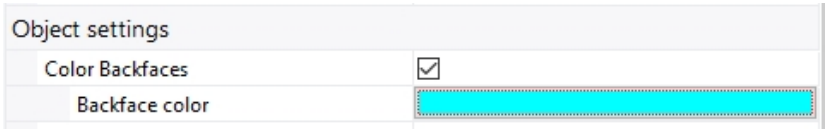


Inside of surface

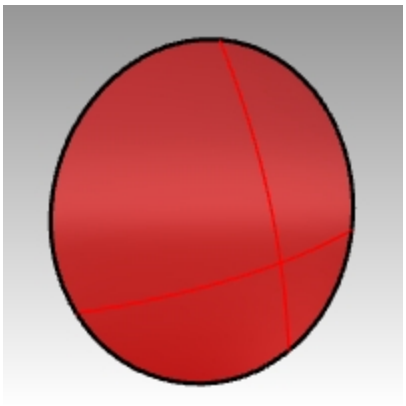
6. On the **Panel** menu, click to open the **Display** panel.
7. On the **Display** panel, next to **Color Backfaces** click the check box.



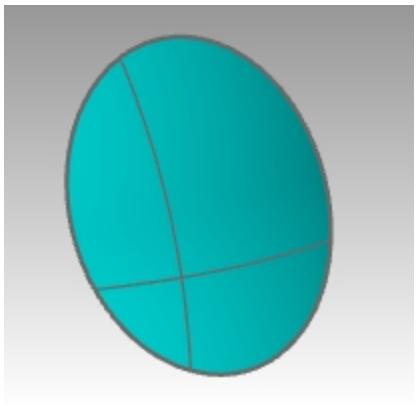
8. Next to **Backface color**, pick the color swatch.



9. In the **Select Color** dialog, pick Cyan color and pick OK.
10. Rotate the view and verify that the backface of the surface is now displayed in cyan.



Outside of surface - same.



Inside of surface - now displays cyan.

Chapter 5 - Curve creation and continuity

This part of the course starts with a review of few concepts and techniques related to NURBS curves that will simplify the learning process during the rest of the class. Curve building techniques have a significant effect on the surfaces that you build from them.

Curve degree

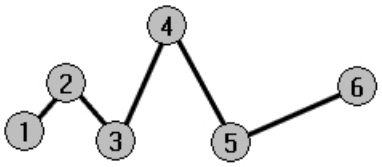
The degree of a curve refers to the highest degree polynomial in the equation for the curve. In practice, it relates to the extent of the influence a single control point has over the length of the curve.

For higher degree curves, a control point has less local influence and a more broad influence over the entire length of the curve. It also has higher internal continuity.

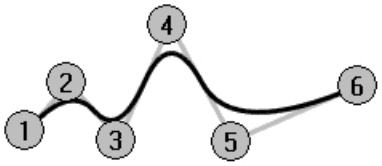
In the example below, the five curves have their control points at the same six points. Each curve has a different degree. The degree can be set with the **Curve** command, Degree option.

Exercise 5-1 Observe curve degree

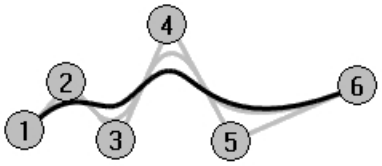
1. **Open** the model **Curve Degree.3dm**.
2. Use the **Curve** command *(Curve menu: Free-Form > Control Points)* with **Degree** set to **1**, using the **Point object snap** to snap to each of the points.



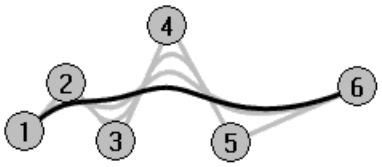
3. Repeat the **Curve** command with **Degree** set to **2**.



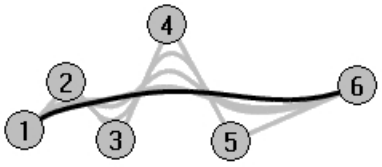
4. Repeat the **Curve** command with **Degree** set to **3**.



5. Repeat the **Curve** command with **Degree** set to **4**.

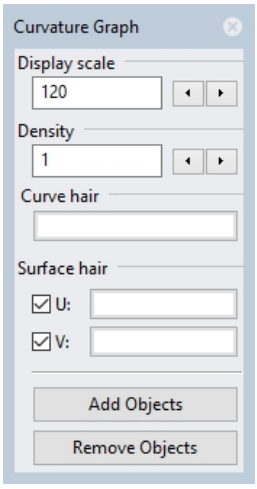


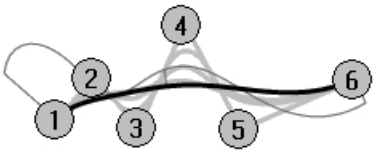
6. Repeat the **Curve** command with **Degree** set to **5**.



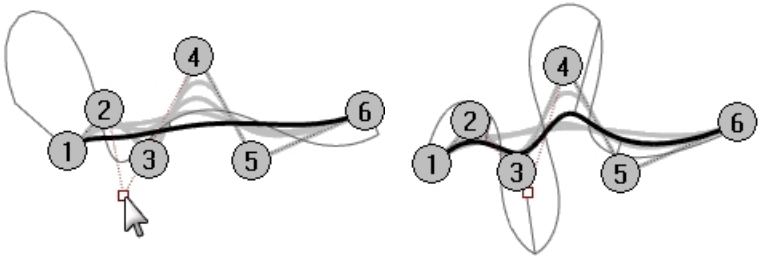
Analyze the curvature of a curve

1. Use the **CurvatureGraph** command *(Analyze menu: Curve > Curvature Graph On)* to turn on the curvature graph for one of the curves.
2. Set the **Display scale** to a number that shows the graph as in the illustration. A number between 110-120 should work well.
   The graph indicates the curvature on the curve. This is the inverse of the radius of curvature. The smaller the radius of curvature at any point on the curve, the larger the amount of curvature.





3. Turn on the control points for the curve you have graphed and view the curvature graph as you drag some control points.
   Note the change in the curvature hairs as you move points.
4. Repeat this process for each of the curves.
   You can use the **Curvature Graph** dialog box buttons to remove or add objects from the graph display.



Note

- **Degree 1** curves have no curvature and no graph displays.
- **Degree 2** curves are internally continuous for tangency. The steps in the graph indicate this condition. Note that only the graph is stepped not the curve.
- **Degree 3** curves have continuous curvature. The graph will not show steps but may show hard peaks and valleys. Again, the curve is not kinked at these places. The graph shows an abrupt but not discontinuous change in curvature.
- In higher degree curves, higher levels of continuity are possible.
  For example, a **Degree 4** curve is continuous in the rate of change of curvature. The graph does not show any hard peaks.
- A **Degree 5** curve is continuous in the rate of change of the rate of change of curvature. The graph does not show any particular features for higher degree curves but it will tend to be smooth.
- Changing the degree of the curve to a higher degree with the **ChangeDegree** command with **Deformable=No** will not improve the internal continuity, but lowering the degree will adversely affect the continuity.
- Rebuilding a curve with the **Rebuild** command will change the internal continuity.

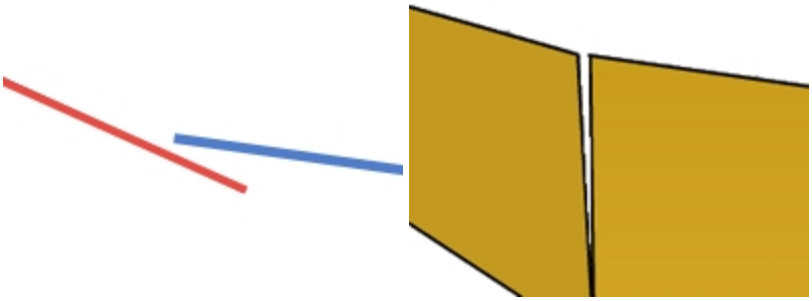
Curve and surface continuity

Since creating a good surface so often depends upon the quality and continuity of the input curves, it is worthwhile clarifying the concept of continuity among curves.

For most curve building and surface building purposes we can talk about four useful levels of continuity:

Not continuous

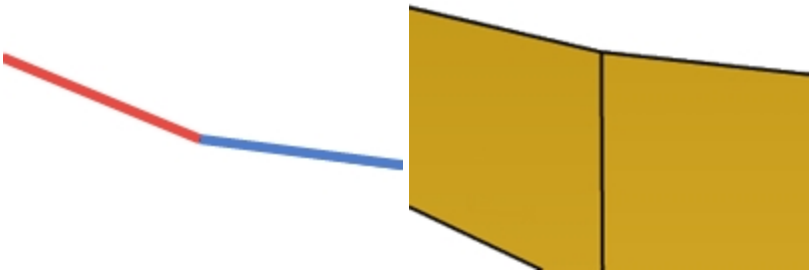
The curves or surfaces do not meet at their end points or edges. Where there is no continuity, the objects cannot be joined.



Positional continuity (G0)

Curves meet at their end points, surfaces meet at their edges.

Positional continuity means that there is a kink at the point where two curves meet. The curves can be joined in Rhino into a single curve but there will be a kink and the curve can still be exploded into at least two sub-curves.

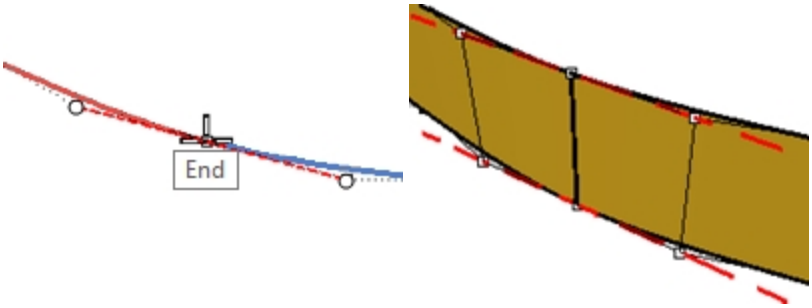


Similarly two surfaces may meet along a common edge but will show a kink or seam, a hard line between the surfaces. For practical purposes, only the endpoints of a curve or the last rows of points along the edges of two untrimmed surfaces need to match to determine G0 continuity.

Tangency continuity (G1)

Curves or surfaces meet and the directions of the tangents at the endpoints or edges is the same. You should not see a crease or a sharp edge.

Tangency is the direction of a curve at any particular point along the curve

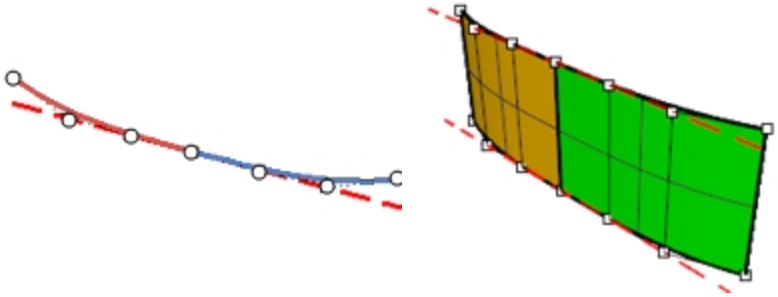


Where two curves meet at their endpoints the tangency condition between them is determined by the direction in which the curves are each heading exactly at their endpoints. If the directions are collinear, then the curves are considered tangent. There is no hard corner or kink where the two curves meet. This tangency direction is controlled by the direction of the line between the end control point and the next control point on a curve.

In order for two curves to be tangent to one another, their endpoints must be coincident (G0) and the second control point on each curve must lie on a line passing through the curve endpoints. A total of four control points, two from each curve, must lie on this imaginary line.

Curvature continuity (G2)

Curves or surfaces meet, their tangent directions are the same and the radius of curvature is the same for each at the endpoint.



Curvature Continuity includes the above G0 and G1 conditions and adds the further requirement that the radius of curvature be the same at the common endpoints of the two curves. Curvature continuity is the smoothest condition over which the user has any direct control, although smoother relationships are possible.

For example, G3 continuity means that not only are the conditions for G2 continuity met, but also that the rate of change of the curvature is the same on both curves or surfaces at the common endpoints or edges.

G4 means that the rate of change of the rate of change of the curvature is the same for both curves where they meet. This is the smoothest type of join. Rhino has tools to build such curves and surfaces, but fewer tools for checking and verifying such continuity than for G0-G2.

G5+ has no visible evidence of more continuity.

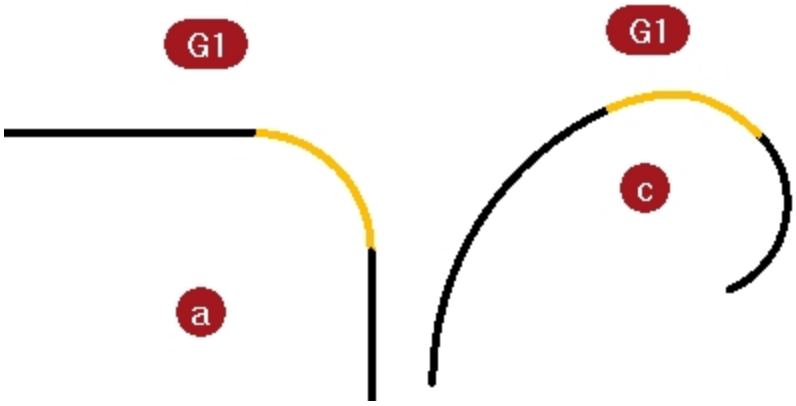
Curve continuity and curvature graph

Rhino has two analysis commands that will help illustrate the difference between curvature and tangency. In the following exercise, we will use the CurvatureGraph and the Curvature commands to gain a better understanding of tangent and curvature continuity.

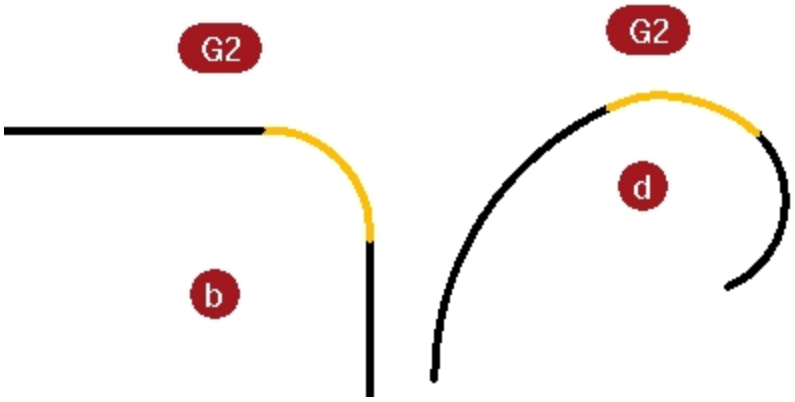
Show continuity with a curvature graph

1. **Open** the model **Curvature_Tangency.3dm**.
   There are five sets of curves, divided into three groups.
   One group has positional (G0) continuity at their common ends.

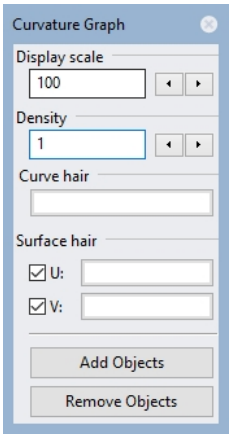
   Group (a-c) has tangency (G1) continuity at their common ends.



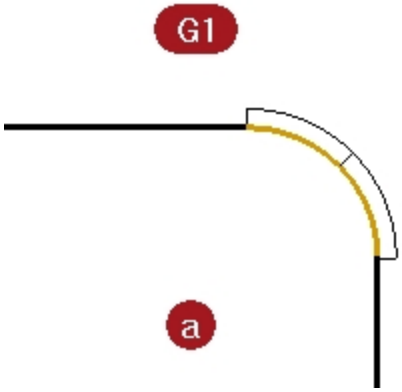
Group (b- d) has curvature (G2) continuity at their common endpoints.



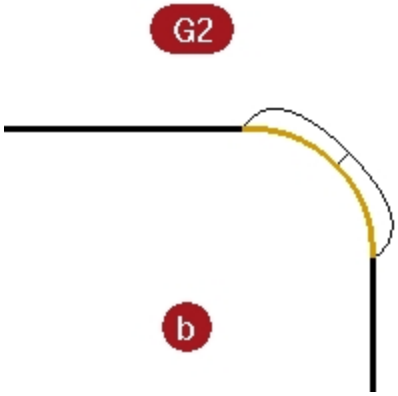
2. Press Ctrl + A to select all of the curves.
3. Turn on the **Curvature Graph** *(Analyze menu: Curve > Curvature Graph On)* for the curves.



4. Set the **Display scale** in the floating dialog box to **100**.
   Change the scale if you cannot see the curvature hair.
   The depth of the graph at this setting shows, in model units, the amount of curvature in the curve.
5. Notice the top sets of curves (**a-b**).
   These two lines have a curve between them.
   Lines have no curvature and therefore do not show a curvature graph.
   The image below shows what happens when the curvature is not continuous. The sudden jump in the curvature graph indicates a discontinuity in curvature.
   The G1 middle curve is an arc. It shows a constant curvature graph as expected because the curvature of an arc never changes, just as the radius never changes.

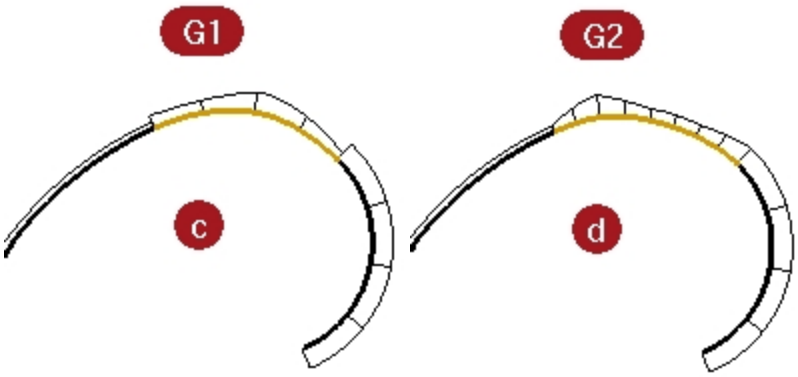


Nevertheless the line-arc-line is smoothly connected. The arc picks up the exact direction of one line and then the next line takes off at the exact direction of the arc at its end.
On the other hand, the G2 curves (b) again show no curvature on the lines, but the curve joining the two lines is different from the G1 case. This curve shows a graph that starts out at zero, it comes to a point at the end of the curve, increases rapidly but smoothly, and then tails off again to zero at the other end where it meets the other straight line segment. It is not a constant curvature curve and thus not a constant radius curve. The graph does not have a sudden jump in the curvature graph; it goes smoothly from zero to its maximum.
For the G2 middle curve, the graph ramps up from zero to some maximum height along a curve and then slopes back to zero, matching the zero curvature again on the other straight line.



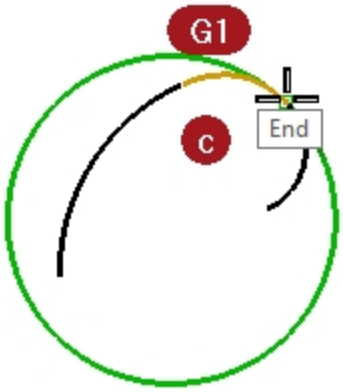
Thus, there is no discontinuity in curvature from the end of the straight line to end of the curve. The curve starts and ends at zero curvature just like the lines have. So, the G2 case not only is the direction of the curves the same at the endpoints, but the curvature is the same there as well. There is no jump in curvature, and the curves are considered G2 or curvature continuous.

6. Look at the **c** and **d** curves.
   These are also G1 and G2 but are not straight lines so the graph shows up on all of the curves.
   Again, the G1 set shows a step up or down in the graph at the common endpoints of the curves. This time the curve is not a constant arc. The graph shows that it increases in curvature out towards the middle.
   On G2 curves, the graph for the middle curve shows the same height as the adjacent curves at the common endpoints. There are no abrupt steps in the graph.
   The outer curve on the graph from one curve stays connected to the graph of the adjacent curve.

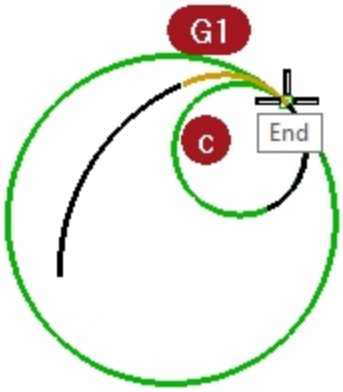


Show continuity with a curvature circle

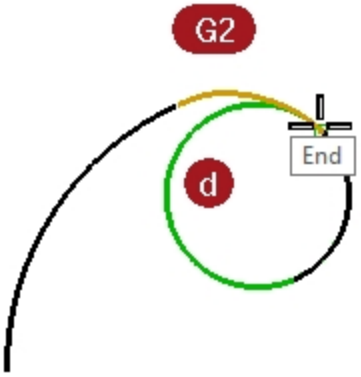
1. Start the **Curvature** command *(Analyze menu: Curvature circle)* and select the middle curve in **set c**.
   The curvature circle that appears on the curve indicates the radius of curvature at that location. The curvature circle results from the center and radius measured at that point on the curve.
2. Drag the curvature circle along the curve.
   Notice that where the circle is the smallest, the graph shows the largest amount of curvature. The curvature is the inverse of the radius at any point.



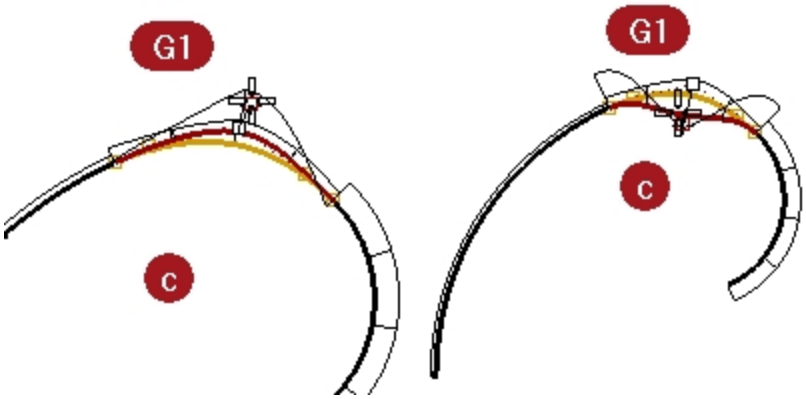
3. On the command-line, click the **MarkCurvature** option.
   Slide the curvature circle, snap to an endpoint of the curve, and click to place a curvature circle.
4. Stop the command and restart it for the other curve sharing the endpoint just picked.
5. Place a circle on this endpoint as well.
   The two circles have greatly different radii. Again, this indicates a discontinuity in curvature. These curves are G1/tangent only, so the curvature at the tangent meeting point is different for the two curves, and that is where the curvature graph would take a jump.



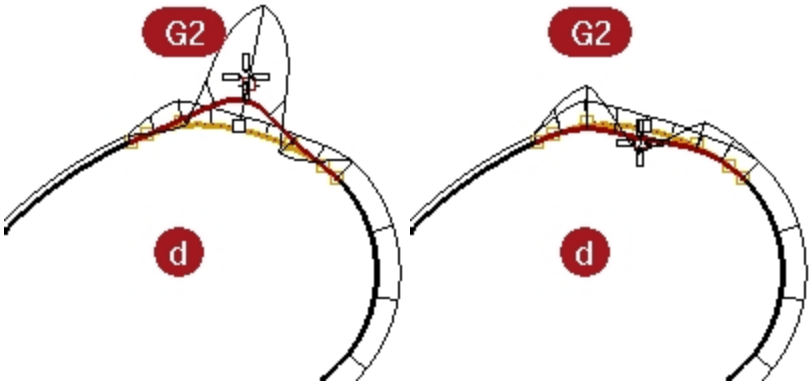
6. Repeat the same procedure to get circles at the ends of the curves in set **d**.
   Notice that this time the circles from each curve at the common endpoint are the same radius. These curves are curvature continuous.



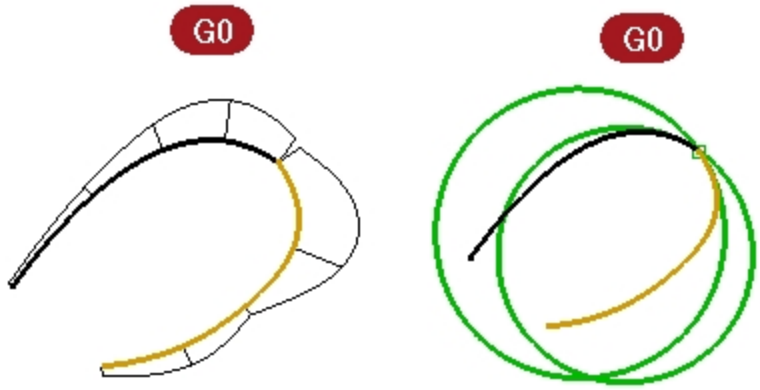
7. Turn on the control points for the middle curves in **c** and **d**.
8. Select the middle control point on either curve and move it around.
   Notice that while the curvature graph changes greatly, the continuity at each end with the adjacent curves is not affected.
   The G1 curve graphs stay stepped, though, the size of the step changes.



   The G2 curve graphs stay connected, although there is a peak that forms there.



9. Look at the graphs for the G0 curves.
   Notice that there is a gap in the graph. This indicates that there is only G0 or positional continuity.
   The curvature circles, on the common endpoints of these two curves, are not only different radii, but they are also not tangent; they cross each other. There is a discontinuity in direction at the ends.

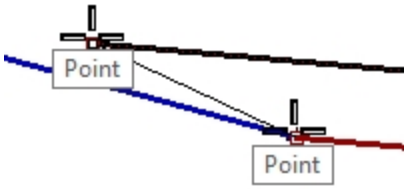


Exercise 5-2 Examine geometric continuity

1. **Open** the model **Curve Continuity.3dm**.
   The two curves are clearly not tangent.
2. Verify this with the continuity checking **GCon** command .
3. Start the **GCon** command *(Analyze menu: Curve > Geometric Continuity)*.
4. Click near the common ends (1 and 2) of each curve.
   Rhino displays a message on the command-line indicating the curves are out of tolerance. The endpoints of the two curves are not close enough to each other to be considered the same.
   Curve end difference = 0.030 millimeters
   Radius of curvature difference = 126.531 millimeters
   Curvature direction difference in degrees = 10.277
   Tangent difference in degrees = 10.277
   Curve ends are out of tolerance.
   Often, imported curves are often out of tolerance and need repair for accurate modeling.

Make curves have position continuity

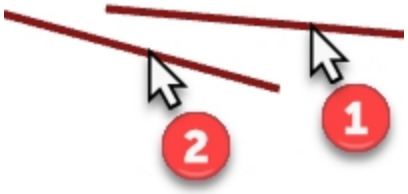
1. Turn on the **control points** for both curves and zoom in on the common ends.
2. Turn on the **Point object snap** and drag one of the endpoints onto the other.
3. Repeat the **GCon** command.
   The command-line message is different now:
   Curve end difference = 0.000 millimeters
   Radius of curvature difference = 126.771 millimeters
   Curvature direction difference in degrees = 10.307
   Tangent difference in degrees = 10.307
   Curves are G0.
4. **Undo** the previous operation.



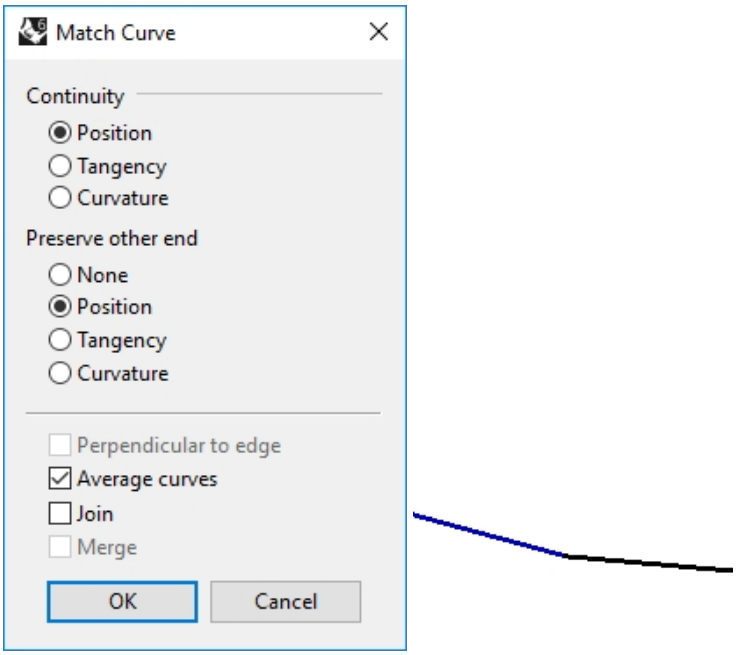
Change curves to position continuity

The **Match** command provides a tool for moving the curve endpoints automatically.

1. Start the **Match** command *(Curve menu: Curve Edit Tools > Match)*.
2. Pick near the common end of one of the curves.
3. Pick near the common end of the other curve.
   By default, the curve you pick first will be the one that is modified to match the other curve.
4. To make both curves change to an average of the two, in the **Match Curve** dialog box, check the **Average Curves** option.



5. In the **Match Curve** dialog box, for **Continuity**, check the **Position** option, for **Preserve other end**, check the **Position** option, check the **Average Curves** option.
6. Repeat the **GCon** command.
   The command-line message indicates:
   Curve end difference = 0.000 millimeters
   Radius of curvature difference = 126.708 millimeters
   Curvature direction difference in degrees = 10.265
   Tangent difference in degrees = 10.265
   Curves are G0.



Set up aliases

Along and **Between** are one-time object snaps that are available in the **Tools** menu under **Object snaps**. They can be used only after a command has been started and apply to only one pick.

Before we continue, we will create some aliases that will be used in the next exercises.

Exercise 5-3 Make Along and Between aliases

1. In the **Rhino Options** dialog box on the **Aliases** page, click the New button

2. In the **Alias** column, type **a**.
   In the **Command macro** column, type **_Along**.
3. In the **Alias** column type **b**.
4. In the **Command macro** column, type **_Between**.
5. **Close** the **Rhino Options** dialog box.

Alias:	Command macro:
a	Along
AdvancedDisplay	! _OptionsPage _AdvancedSettings
b	Between

Tangent continuity

It is possible to establish a tangency (G1) condition between two curves by aligning the control points in a particular way. The endpoints at one end of the curves must be coincident and these points in addition to the next point on each curve must fall in a line with each other. This can be done automatically with the **Match** command, although it is also easy to do by moving the control points using the normal Rhino transform commands.

We will use **Move**, **SetPt**, **Rotate**, **Zoom Target**, **PointsOn** F10, **PointsOff** F11 commands and the object snaps **End**, **Point**, **Along**, **Between** and the Tab lock to move the points in various ways to achieve tangency.

Tab direction lock

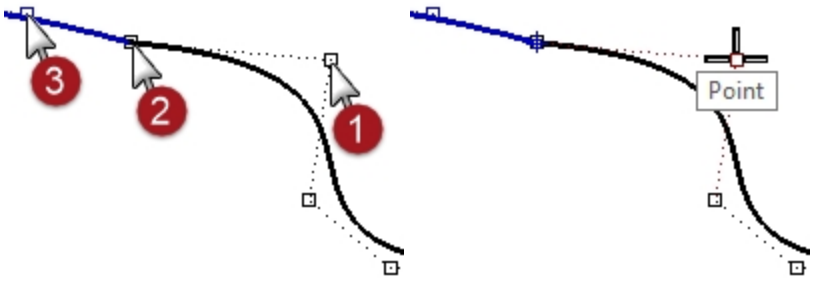
The Tab direction lock locks the movement of the cursor when the Tab key is pressed. It can be used for moving objects, dragging, or curve and line creation.

To activate Tab direction lock press and release the Tab key when Rhino is asking for a location in space. The cursor will be constrained to a line between its location in space at the time the Tab key is pressed and the location in space of the last clicked point.

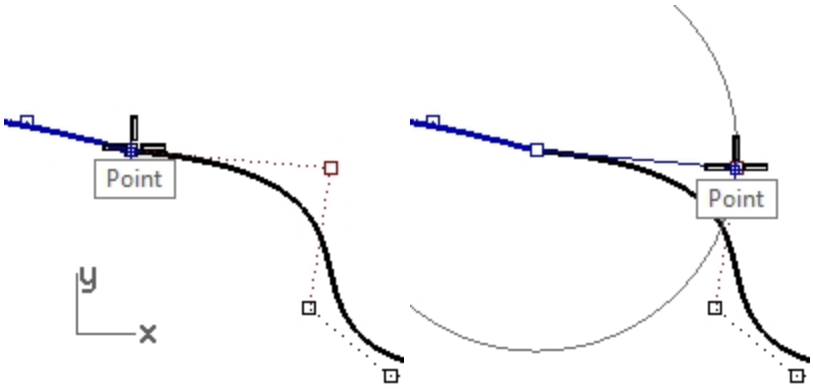
When the direction is locked, it can be released with another press and release of the Tab, and a new, corrected direction set with yet another Tab press.

Change the continuity using Rotate and the Tab direction lock

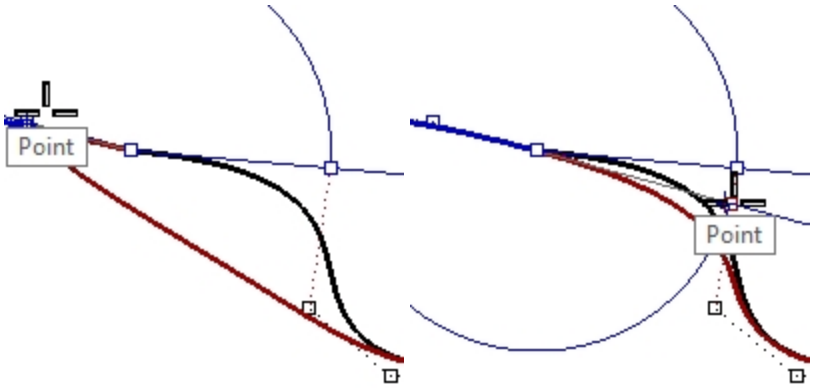
1. Turn on the control points for both curves.
2. Select the control point (1) second from the end of one of the curves.
3. Start the **Rotate** command *(Transform menu: Rotate)*.
4. Using the **Point object snap**, select the common endpoints (2) of the two curves for the **Center of rotation**.
5. For the **First reference point**, snap to the current location of the selected control point.



6. For the **Second reference point**, make sure the point object snap is still active.
   Hover the cursor, but do not click, over the second point (3) on the other curve.
   While the **Point** object snap flag is visible on screen, indicating the cursor is locked onto the control point, press and release the Tab key.
   Do not click with the mouse.

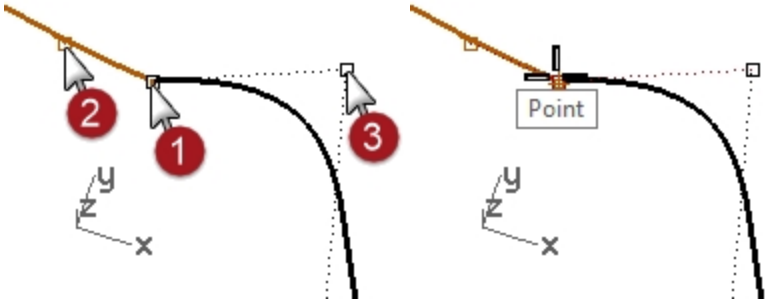


7. Bring the cursor back over to the other curve.
   Notice that the position is constrained to a line between the center of rotation and the second point on the second curve; that is, the location of the cursor when you press the Tab key. You can now click the mouse on the side opposite the second curve.
   During rotation, the Tab direction lock knows to make the line from the center and not from the first reference point.
   The rotation endpoint will be exactly in line with the center of rotation and the second point on the second curve.

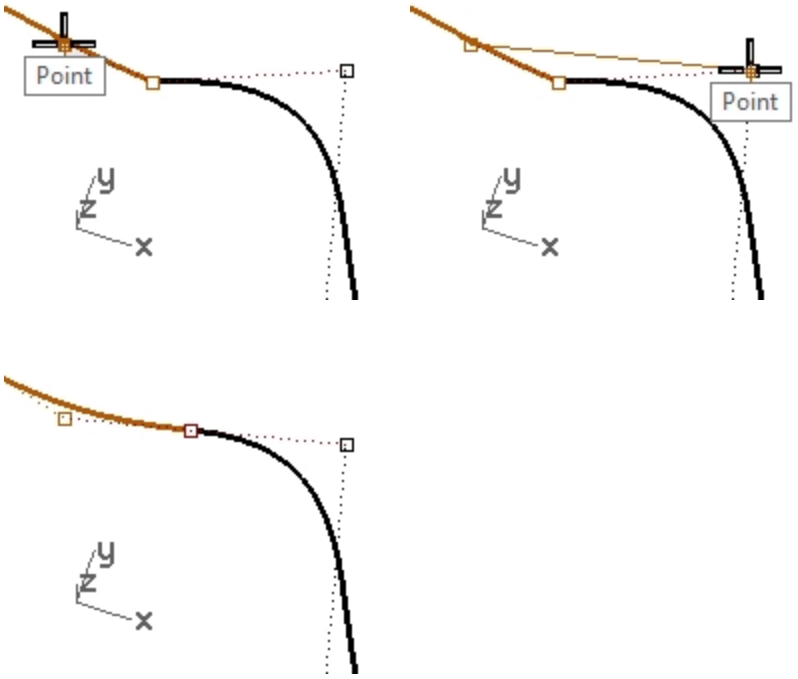


Change the continuity by adjusting control points

1. Use the **OneLayerOn** command to turn on only the **3D Curves** layer.
2. Check the continuity of the curves with the **GCon** command.
3. Turn on the control points for both curves.
4. Window select the common endpoints of both curves (1).
5. Use the **Move** command *(Transform menu: Move)* to move the points.



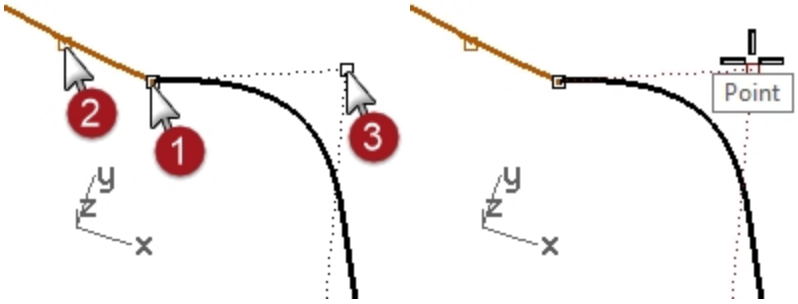
6. For the **Point to move from**, snap to the same point (1).
7. For the **Point to move to**, type **b** and press Enter to use the **Between** object snap.
8. For the **First point**, snap to the second point (2) on one curve.
9. For the **Second point**, snap to the second point (3) on the other curve.
   The common points are moved in-between the two second points, aligning the four points.



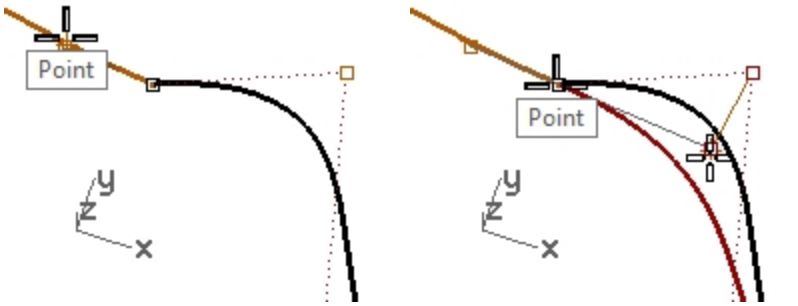
10. Check the continuity.

Change the continuity by adjusting control points using the Along object snap

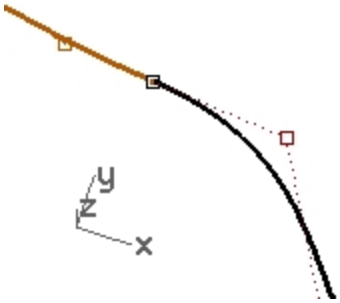
1. **Undo** the previous operation.
2. Select the second point (3) on the curve on the right.
3. Use the **Move** command *(Transform menu: Move)* to move the point.
4. For the **Point to move from**, snap to the selected point.



5. For the **Point to move to**, type **A** and press Enter to use the **Along object snap**.
6. For the **Start of tracking line**, snap to the second point (2) on the other curve.
7. For the **End of tracking line**, snap to the common points (1).
   The point tracks along a line that goes through the two points, aligning the four points.



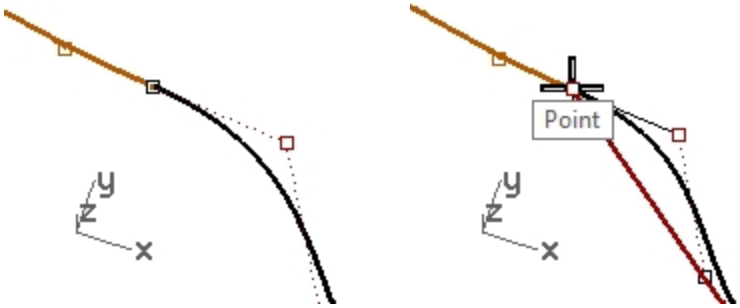
8. Click to place the point.
9. Check the continuity.



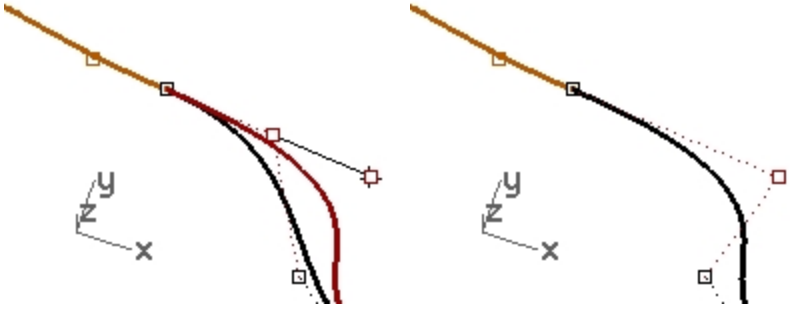
Edit the curves without changing the tangency direction

With the Tab technique, we can adjust the meeting point of the curves, or the shape of either curve near the meeting point, without losing the G1 continuity.

1. Window select the common endpoints or select the second point on either curve.
   Turn on the **Point** object snap and drag the point(s) to the next one of the four critical points.
2. When the **Point** object snap flag shows on the screen, use the Tab direction lock by pressing and releasing the Tab key without releasing the mouse button.



3. Drag the point(s) and the tangency is maintained since the drag direction is constrained to the **Tab** direction lock line.
4. Release the left mouse button at any point to place the point(s).



Note

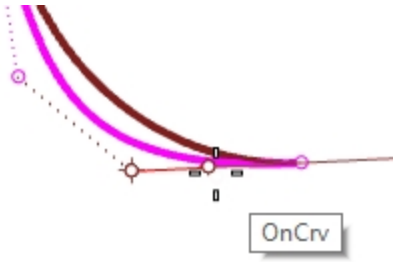
- To maintain G1 continuity make sure that any point manipulation of the critical four points takes place along the line on which they all fall.
- Once you have G1 continuity you can still edit the curves near their ends without losing continuity, using the Tab direction lock.
- This technique only works after tangency has been established.

Edit the curves using Dragmode

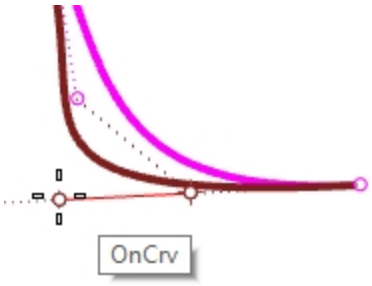
1. Start the **DragMode** command, and choose **ControlPolygon** at the command line
   Note: the cursor changes to indicate the drag mode has changed from the default CPlane based dragging.

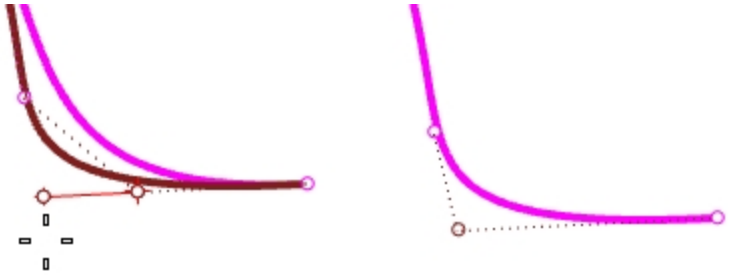
   ControlPolygon only applies to curve and surface control points.
2. Disable **Osnap** in the Status bar.
3. Select the second point from one end of the curve and drag it towards the end point - the dragging is constrained to the control polygon line connecting the points. This ensures that the tangent direction on the curve does not change.

4. Now drag the point to the left and away from the end point on the curve that is on the right.
   The point is constrained to the control polygon of the third point. This control edit breaks that tangency direction of the curve.

5. To maintain the tangent direction of the curve, drag towards the end point a short distance first, hit **Tab** to constrain that direction and then drag away from the end point. Dragging a point with the **Tab** constrains the direction to the curve or surface normal.

6. Run DragMode again to switch back to CPlane drag mode.

The Dragmode Macros

Running the same drag mode option twice in a row returns drag mode to the default, so you can make a shortcut or alias for:

```
! _DragMode _ControlPolygon
! _DragMode _Cplane
```

Use these to toggle back and forth quickly between default and control polygon modes.

Go to **Options** and **Keyboard** and add the **Macro** to keys **Control +F6** and **Control +F7**.

Ctrl+F1	'_SetMaximizedViewport Top
Ctrl+F2	'_SetMaximizedViewport Front
Ctrl+F3	'_SetMaximizedViewport Right
Ctrl+F4	'_SetMaximizedViewport Perspective
Ctrl+F5	! '_historypurge 'seldim _enter
Ctrl+F6	! _DragMode _ControlPolygon
Ctrl+F7	! _DragMode _Cplane

Note: You will find the macro command in the **Macros.txt** which is included in the Level 2 model set.

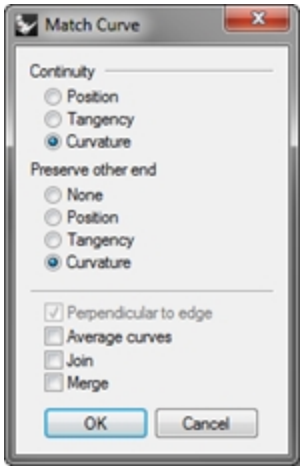
Curvature continuity

Adjusting points to establish curvature continuity is not as straightforward as for tangency. Curvature at the end of a curve is determined by the position of the last three points on the curve, and their relationships to one another are not as straightforward as it is for tangency.

To establish curvature or G2 continuity, the **Match** command is the only practical way in most cases.

Exercise 5-4 Match the curves

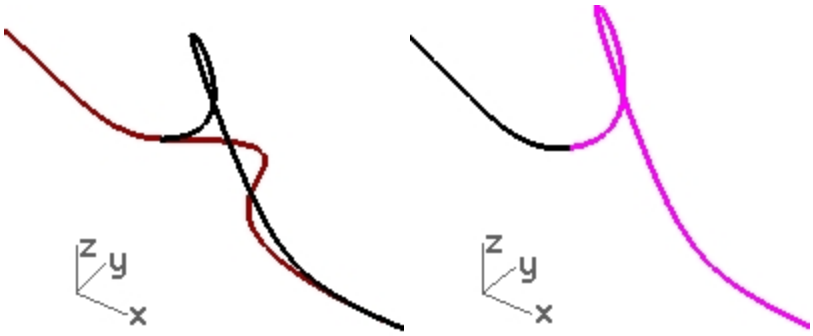
1. Use the **Match** command *(Curve menu: Curve Edit Tools > Match)* to match the magenta (1) curve to the red (2) curve.
2. Set **Continuity** to **Curvature**, **Preserve other end** to **Curvature**, and clear the **Average curves** option.



When you use **Match** with **Curvature** checked on these particular curves, the third point on the curve to be changed is constrained to a position calculated by Rhino to establish the desired continuity.

The curve being changed is significantly altered in shape.

Moving the third point by hand will break the G2 continuity at the ends, though G1 will be maintained

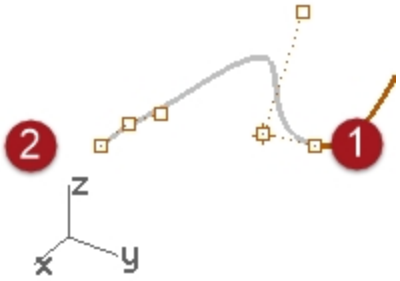


Advanced techniques for controlling continuity

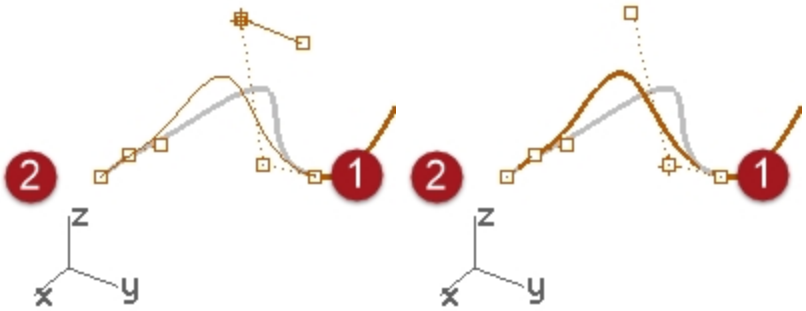
There are two additional methods to edit curves while maintaining continuity in Rhino. (1) The **EndBulge** command constrains points at the end to maintain continuity with the adjacent curve. (2) Adding knots will allow more flexibility when changing the curve's shape.

Edit the curve with end bulge

1. **Right-click** the **Copy** button to make a duplicate of the magenta curve and then **Lock** it.
2. Start the **EndBulge** command *(Edit menu: Adjust End Bulge)*.
3. Select the magenta curve.
   Notice that there are more points displayed than were on the original curve.
   The **EndBulge** command adds more control points to the curve if the curve has less than the required control point count.



4. Select the third point, drag it, and click to place the point, press Enter to exit the command.
   If the endpoint of the curve has G2 continuity with another curve, the G2 continuity will be preserved, because **EndBulge** preserves the curvature at the endpoint of the curve.



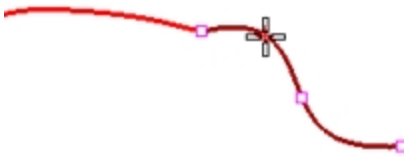
Note: Adjusting control points will work to match curvature only in the simple case of matching to a straight line.

Add a knot

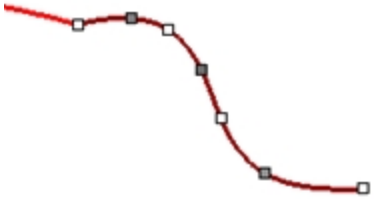
Adding a knot or two to the curve will put more points near the end so that the third point can be nearer the end. Knots are added to curves and surfaces with the **InsertKnot** command.

1. **Undo** your previous adjustments.

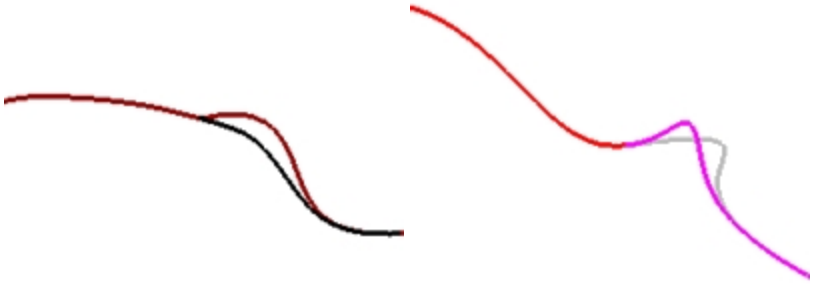
2. Start the **InsertKnot** command *(Edit menu: Control Points > Insert Knot)*.
3. Select the magenta curve.
4. Pick a location on the curve to add a knot in between the first two knot markers.
   In general, a curve or surface will tend to behave better in point editing if new knots are placed midway between existing knots, thus maintaining a uniform distribution.
   Adding knots also results in added control points.



   Knots and control points are not the same thing and the new control points will not be added at exactly the new knot location.
   The **Automatic** option automatically inserts a new knot in each span exactly half way between existing knots.
   If you only want to place knots in some of the spans, you should place these individually by clicking on the desired locations along the curve.
   Existing knots are highlighted in white.



5. **Match** the curves after inserting a knot into the magenta curve.
   Inserting knots closer to the end of curves will change how much **Match** changes the curve.



Chapter 6 - Surface continuity

The continuity characteristics for curves can also be applied to surfaces. Instead of dealing with the endpoint, second, and third points, entire rows of points at the edge, and the next two positions away from the edge are involved. The tools for checking continuity between surfaces are different from the simple **GCon** command.

Analyze surface continuity

Rhino takes advantage of the OpenGL display capability to create false color displays for checking curvature and continuity within and between surfaces. These tools are located in the **Analyze** menu, under **Surface**. The tool, which most directly measures G0-G2 continuity between surfaces, is the **Zebra** command. Zebra analysis simulates reflection of a striped background on the surface.

Note: An OpenGL graphics accelerator card is not necessary to use these tools, although they may work faster with OpenGL acceleration.

Match surface continuity

The command used to establish G0, G1 or G2 continuity between surfaces is the **MatchSrf** command.

Match surface options

Option	Description
Average surfaces	Both surfaces are modified to an intermediate shape.
Refine match	Determines if the match results should be tested for accuracy and refined so that the faces match to a specified tolerance.
Match edges by closest points	The surface being changed is aligned to the edge it is being matched to by pulling each edge point to the closest point on the other edge.
Preserve other end	If the surface does not have enough points its degree is raised (up to a maximum of 5), until there are enough points.

Isocurve direction adjustment

Specifies the way the parameterization of the matched surfaces is determined.

Option	Description
Automatic	Evaluates the target edge, then uses Match target isocurve direction if it is an untrimmed edge or Make perpendicular to target edge if it is a trimmed edge.
Preserve isocurve direction	As closely as possible, keeps the existing isocurve directions the same as they were in the surface before matching.
Match target isocurve direction	Makes the isocurves of the surface that is being adjusted parallel to those of the surface it matches.
Make perpendicular to target edge	Makes the isocurves of the surface that is being adjusted perpendicular to the edge being matched.

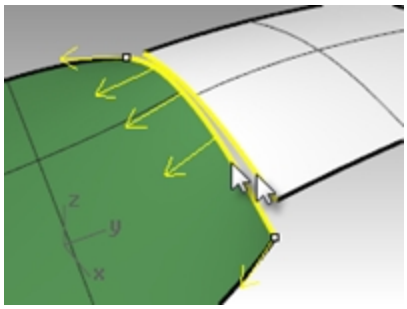
Surface continuity and MatchSrf

The **MatchSrf** command takes surface edges as input and modifies one or both of the surfaces. You need to tell the command exactly which edge to change and then which edge to match to the target surface. We will match the white surface's edge to the green one first. Both the edge to change and the edge to match to are untrimmed on these surfaces.

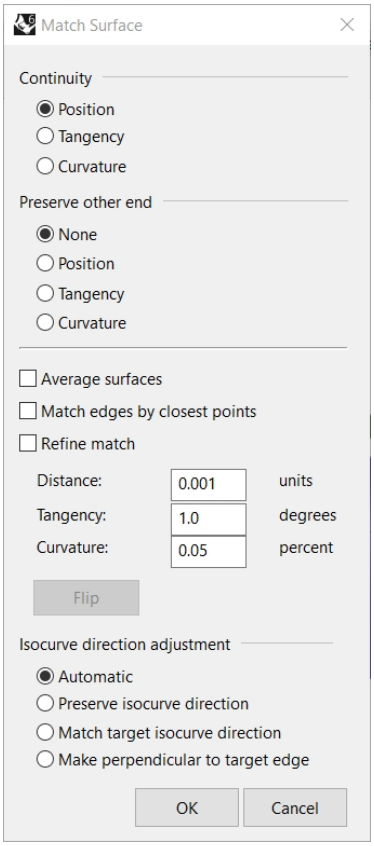
While **MatchSrf** is generally used to adjust surfaces that are fairly close to being at the desired continuity, this example is somewhat exaggerated in order to clearly show the functionality and options.

Exercise 6-1 Practice matching surface continuity

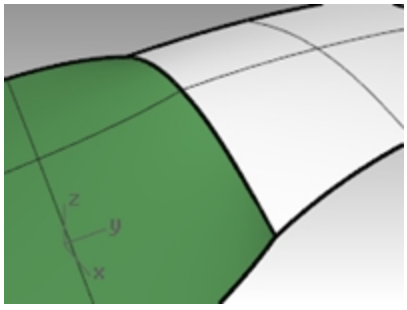
1. Open the model **Surface Continuity.3dm**.
2. Start the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)*.
3. Select the edge of the white surface on the edge nearest the green surface.
4. Select the edge of the green surface near the same location as the selection point on the white surface edge and press Enter.



5. In the **Match Surface** dialog box, choose **Position** as the desired **Continuity**, choose **None** for **Preserve other end**, clear the **Refine match** options, and for **Isocurve direction adjustment**, choose **Automatic**.
   Make sure all other check boxes are unchecked.



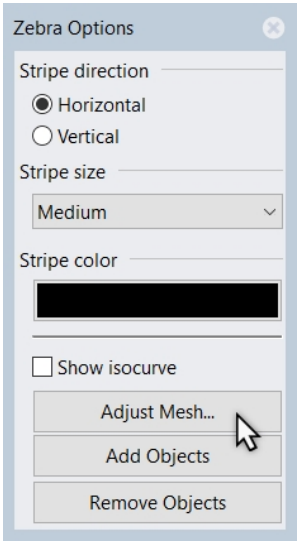
6. A shaded preview is automatically generated so you can see what the result will be like.
7. Click OK.
   The edge of the white surface is pulled over to match the edge of the green one.

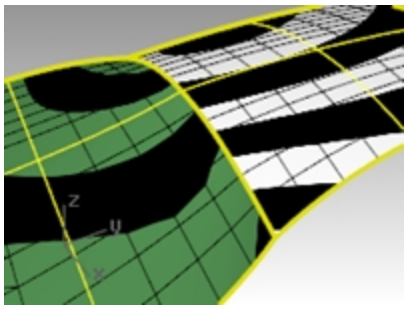


Check the continuity with Zebra analysis

1. Check the surfaces with **Zebra** analysis tool *(Analyze menu: Surface > Zebra)*.
   This command relies on an approximation of the surface for its display information.
   By default, the mesh generated by **Zebra** may be too coarse to get a good analysis of the surfaces. If the display

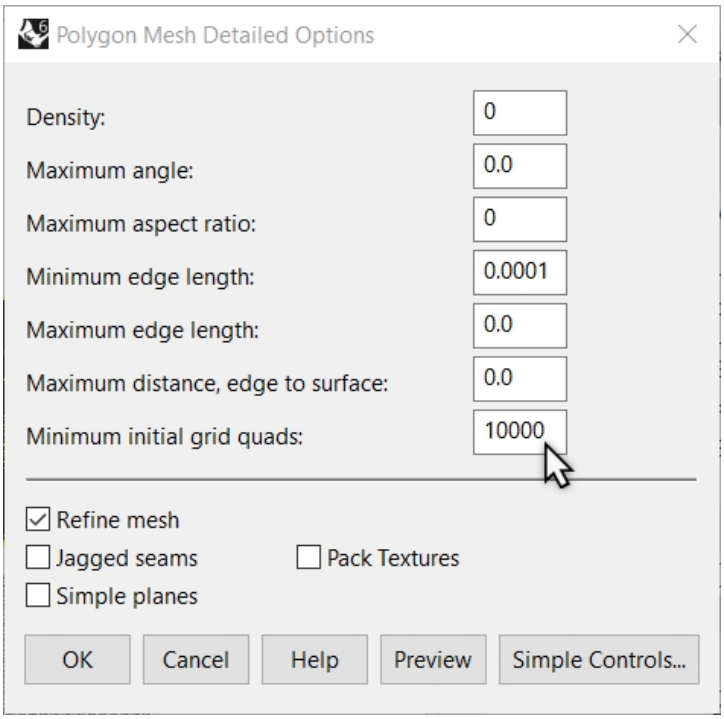
shows very angular stripes rather than smooth stripes on each surface, click the Adjust mesh button on the **Zebra Options** dialog box.



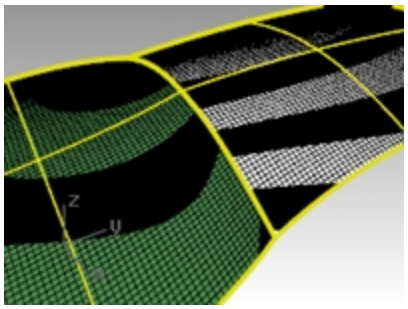


In general, the analysis mesh should be much finer than the normal shade and render mesh meshes.
It is a good habit to set these meshes the first time you use a surface analysis display mode in a model. This setting is then saved in the file.

2. Use the **Detailed Options** to set mesh parameters.

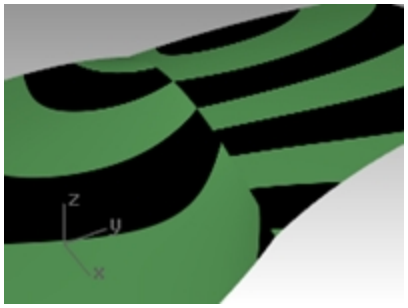


For this type of mesh, it is often easiest to zero out (disable) the Maximum angle setting and rely entirely on the Minimum initial grid quads setting. This number can be quite high but may depend upon the geometry involved.



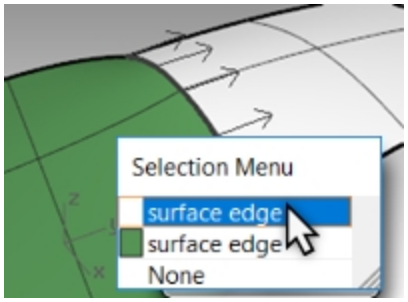
In this example, a setting here of 5000 to 10000 will generate a very fine and accurate mesh.
The analysis can be further improved by joining the surfaces to be tested.

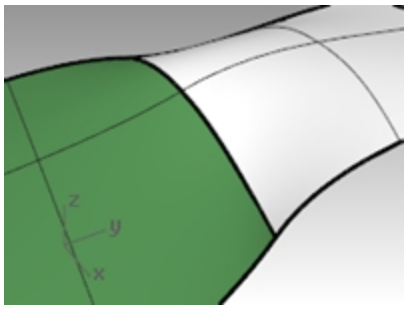
3. **Join** the two surfaces.
   This will force a refinement of the mesh along the joined edge and help the Zebra stripes act more consistently.
   There is no particular correlation between the stripes on one surface and the other except that they touch, indicating G0 continuity.
4. **Undo** the **Join**.



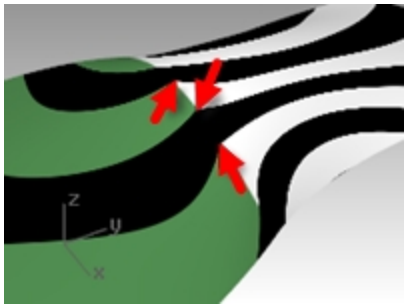
Match the surface to tangency

1. Use the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)* again with the **Tangency** option for **Continuity**.
   When you pick the edge to match you will get direction arrows that indicate which surface edge is being selected. The surface that the direction arrows are pointing toward is the surface whose edge is selected.



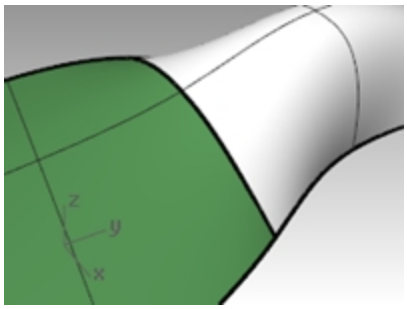


2. Check the surfaces with **Zebra** analysis.
3. Rotate the view to view along the seam.
   The ends of the stripes on each surface meet the ends on the other cleanly, though at an angle.
   This indicates G1 continuity.

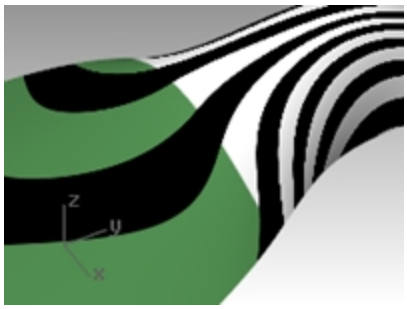


Match the surface to curvature

1. Use the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)* with the **Curvature** option.



2. Check the surfaces with **Zebra** analysis.
   The stripes now align themselves smoothly across the seam. Each stripe connects smoothly to the counterpart on the other surface.
   This indicates Curvature (G2) continuity.



   Note: Doing these operations one after the other may yield different results than going straight to Curvature without first using Position. This is because each operation changes the surface near the edge, so the next operation has a different starting surface.

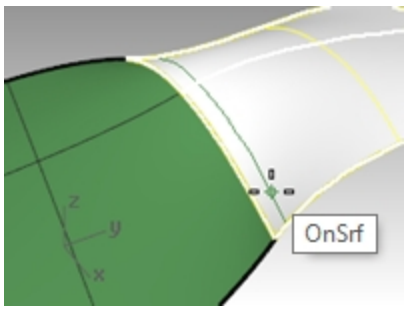
Add knots to control surface matching

As in matching curves, MatchSrf will sometimes distort the surfaces more than is acceptable in order to attain the desired continuity. We will add knots to surfaces to limit the influence of the MatchSrf operation. The new second and third rows of points will be closer to the edge of the surface.

Surfaces can also be adjusted with the **EndBulge** command.

Add a knot to a surface

1. **Undo** the previous operation.
2. Use the **InsertKnot** command *(Edit menu: Control Points > Insert Knot)* to insert a row of knots near the end of the white surface.
   When this command is used on a surface, it has more options.
   You can choose to insert a row of knots in the U-direction, the V-direction, or both.
   Choose **Symmetrical** to add knots at opposite ends of a surface.
3. Use **MatchSrf** to curvature match the surface to the other.
   Notice that the new matched surface is different from the old one.



Use EndBulge to edit the surface shape

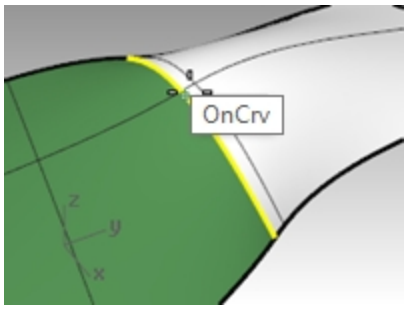
The **EndBulge** command lets you edit the shape of a surface without changing the tangent direction and the curvature at the edge of the surface. This is useful when you need to alter the shape of a surface that has been matched to another surface.

EndBulge allows you to move control points at a specified location on the surface. These points are constrained along a path that keeps the direction and curvature from changing.

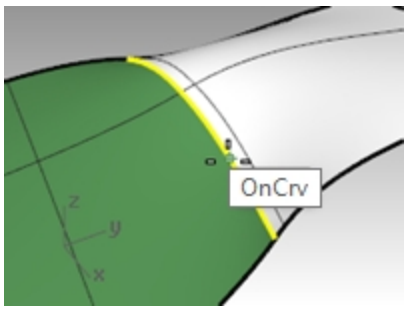
The surface can be adjusted equally along the entire selected edge or along a section of the edge. In the latter case, the adjustment takes place at the specified point and tapers out to zero at either end of the range. Either the start or endpoint of the range can be coincident with the point to adjust, thus forcing the range to be entirely to one side of the adjustment point.

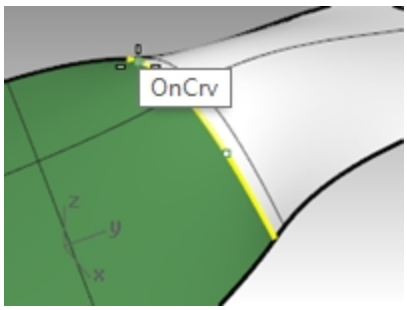
Adjust the surface using end bulge

1. Start the **EndBulge** command *(Edit menu: Adjust End Bulge)*.
2. For the surface edge to adjust, pick the edge of the surface on the right.
3. For the **Point to edit**, pick a point on the edge at which the actual adjustment will be controlled.
   You can use object snaps and reference geometry to select a point with precision.

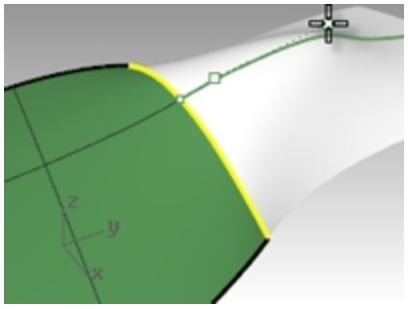


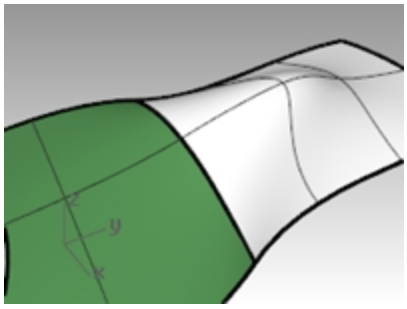
4. For the **Start of region to edit**, pick a point along the common edges to define the region to be adjusted.
5. For the **End of region to edit**, pick another point to define the region to be adjusted.
   To select a range at this point, slide the cursor along the edge and click at the beginning and endpoints of the range. If the whole edge is to be adjusted equally, press Enter.





6. For the **Point to adjust**, select one of the points that are displayed.
   Rhino shows three points, of which you are allowed to manipulate only two. When you move the second point, Rhino also moves the third point that is not being directly manipulated in order to maintain the continuity. If you move the third point it won't change the second point
7. Drag the point and click to adjust the surface.





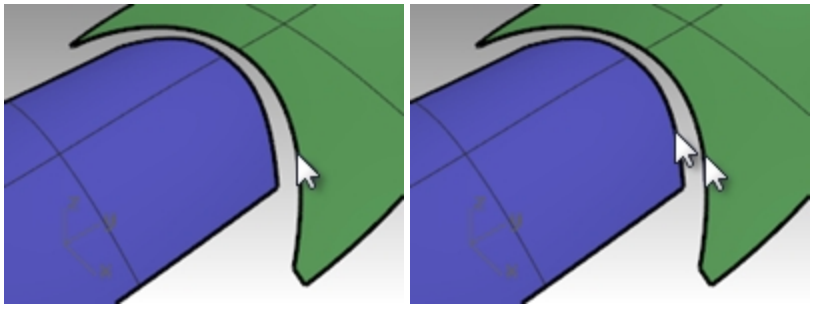
If maintaining the G2 curvature-matching condition at the edge is not needed, use the Continuity=Tangency option to turn off one of the two points available for editing. Only G1 will be preserved.

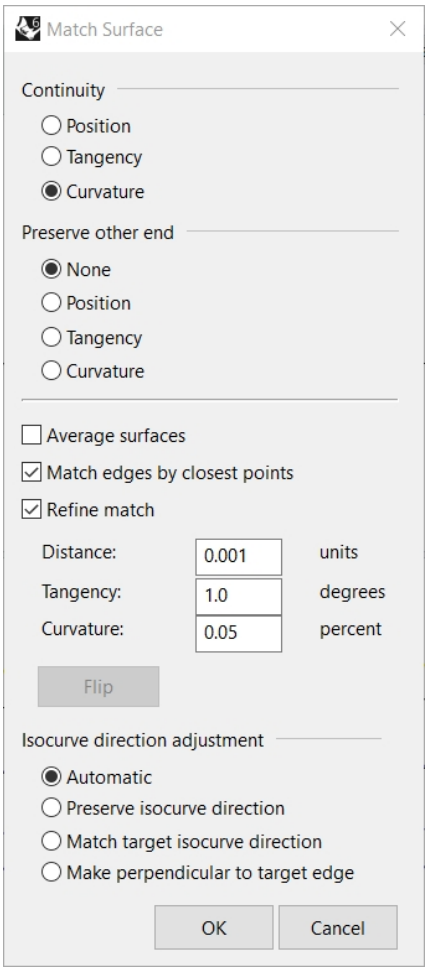
8. Press Enter to end the command.

Match surfaces

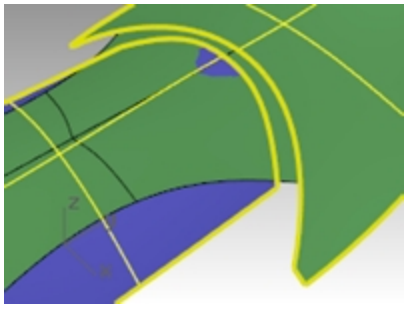
The **MatchSrf** command does not allow matching a trimmed edge to an untrimmed edge. You must work from the untrimmed edge to the trimmed edge.

Match an untrimmed surface to a trimmed surface

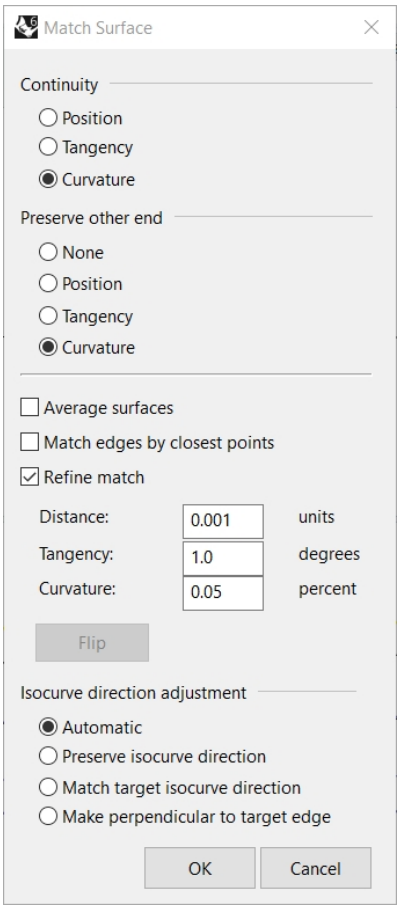
1. Start the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)*.
2. Select the edge of the green surface on the edge nearest the blue surface.
   The edge will not select and you will see the following message on the command-line:
   ```
   Edge must be on the edge of a surface (not a trimmed edge).
   Select an untrimmed surface edge to change (MultipleMatches).
   ```
   
3. Instead, select the untrimmed edge of the blue surface on the edge nearest the green surface.
   Select the trimmed edge of the green surface near the same location as the selection point on the blue surface edge.
4. In the **Match Surface** dialog box, choose **Curvature** as the desired **Continuity**, choose **None** for **Preserve other end**, check the **Match edges by closest points** option, and for **Isocurve direction adjustment** choose **Automatic**.
   Make sure all other check boxes are cleared.



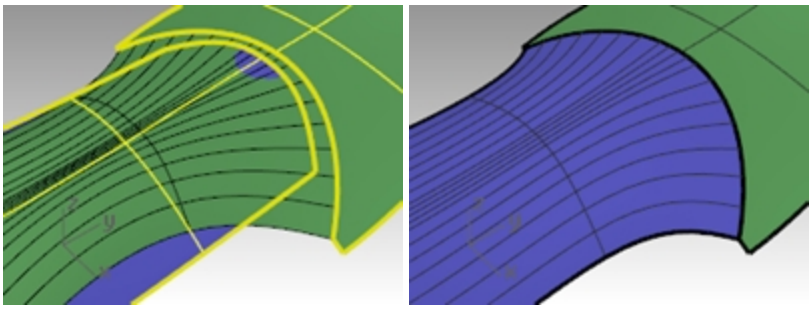
A preview is automatically generated so you can see what the result will be like.
Notice that the blue surface does not include the entire untrimmed edge of the green surface. It only extends as far as the closest point from the original surface.



5. In the **Match Surface** dialog box, clear the **Match edges by closest points** option, and check the **Refine match** option.
6. Toggle through the **Isocurve direction adjustment** and the **Preserve other end** options to see what happens to the matched surface.



7. When finished click OK.



Surfacing commands that pay attention to continuity

Rhino has several commands that can build surfaces using the edges of other surfaces as input curves. They can build the surfaces with G1 or G2 continuity to those neighboring surfaces. The commands are:

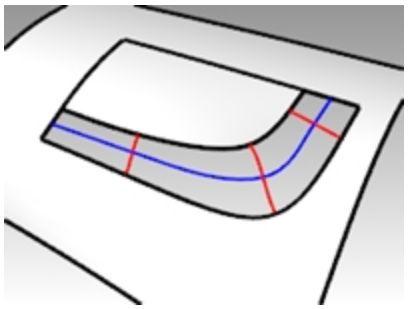
- NetworkSrf
- Sweep2
- Patch (G1 only)
- Loft (G1 only)
- BlendSrf (G1 to G4)

The following exercises will provide a quick overview of these commands.

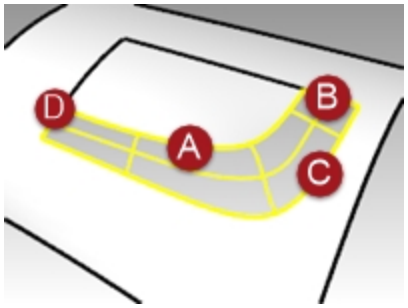
Exercise 6-2 Create a surface from a network of curves

1. Open the model **Continuity Commands.3dm**.
   On the **Surfaces** layer, two joined surfaces have been trimmed leaving a gap. This gap needs to be closed up with continuity to the surrounding surfaces.
2. Turn on the **Network** layer, if it is not already on, and make it current.
   Several curves already in place define the required cross sections of the surface.

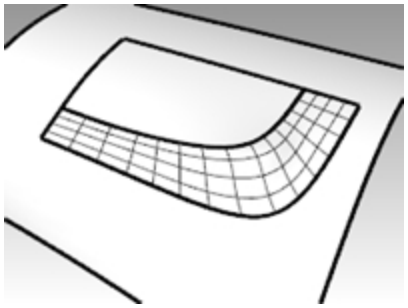
3. Use the **NetworkSrf** command *(Surface menu: Curve Network)* to close the hole with an untrimmed surface using the curves and the edges of the surfaces as input curves.

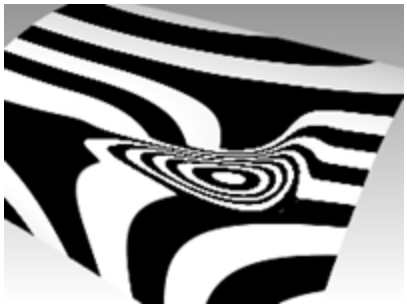


4. For the **Select curves in network** option, select the four edges that border the opening and the four curves inside the opening and press Enter.
   Note that there is a maximum of four edge curves as input. You can also specify the tolerances or maximum deviation of the surface from the input curves.
   By default, the edge tolerances are the same as the model's **Absolute Tolerance** setting. The interior curves' tolerance is set 10 times looser than that by default.



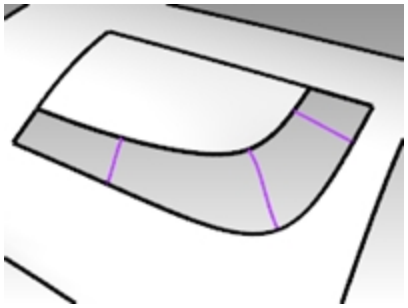
5. In the **Surface From Curve Network** dialog box, choose **Curvature continuity** for all the edges, and click OK.
   The surface that is created has curvature continuity on all four edges.
6. Check the resulting surface with **Zebra** analysis.

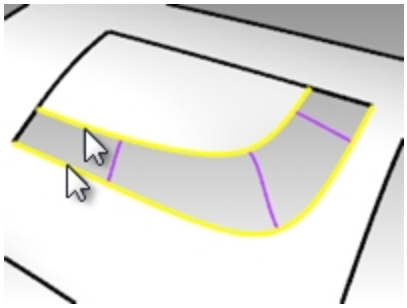




Make the surface with a two-rail sweep

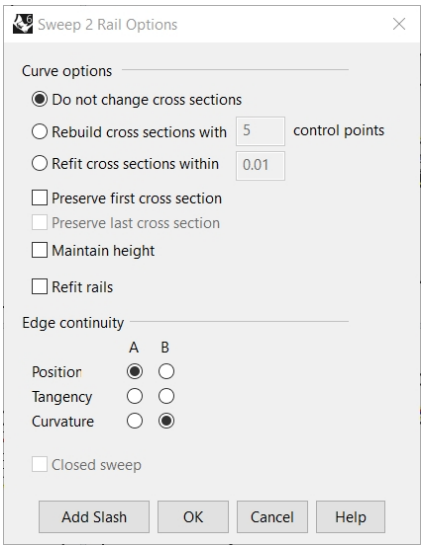
1. Use the **OneLayerOn** command to open the Surfaces layer by itself again, and then click in the layers panel of the status bar, and select the **Sweep2** layer.
2. Start the **Sweep2** command *(Surface menu: Sweep 2 Rails)* and select the long surface edges as the rails.



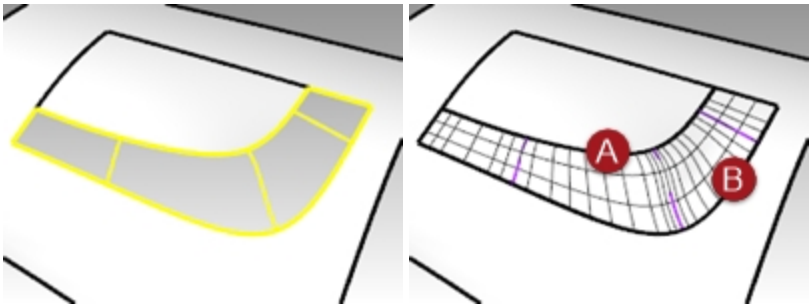


3. Select one short edge, the cross-section curves, and the other short edge as profiles.

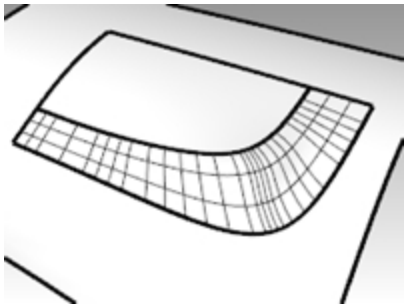
4. Choose **Curvature** for both **Rail curve** options.



Since the rails are surface edges, the display labels the edges, and the **Sweep 2 Rails Options** dialog box gives the option of maintaining continuity at these edges.



5. Click OK.
6. Check the resulting untrimmed surface with **Zebra** analysis.



Create a patch surface

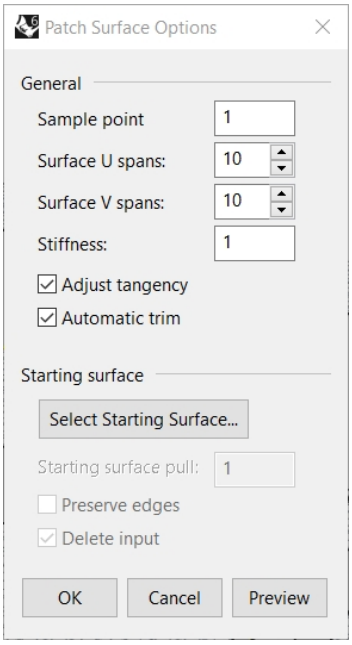
The **Patch** command builds a trimmed surface, if the bounding curves form a closed loop. Patch can match continuity to G1 if the bounding curves are edges. The **Patch** command:

- Can use unlimited curves or points for input
- Ignores noise of many control points
- Is good for scanned data
- Is good for reverse engineering

Make a patch surface

1. Turn on the **Surfaces** and **Patch** layers.
2. Turn all other layers off.
3. Start the **Patch** command *(Surface menu: Patch)*.
4. Select the edge curves and the interior curves, and then press Enter.

5. In the **Patch Surface Options** dialog box, set the following options:
   Set **Sample point spacing** to **1.0**.
   Set **Stiffness** to **1**.
   Set **Surface U** and **V spans** to **10**.
   Check the **Adjust tangency** and **Automatic trim** options, and click OK.



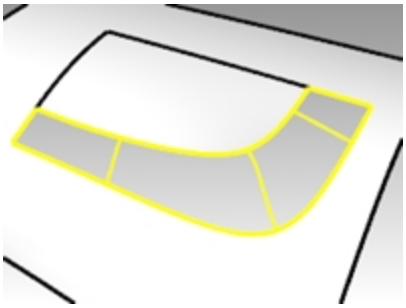
Sample point spacing sets the nominal distance along the input curve between sample points. The minimum is eight points per curve.

Surface U and V spans value sets spans for the patch surface. The default value is 10 spans for both the U & V direction.

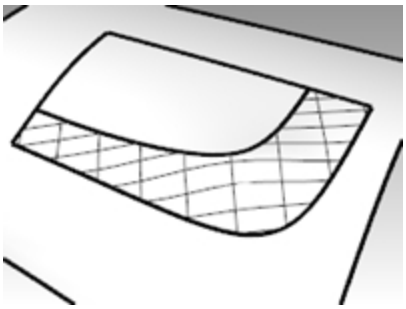
Stiffness value is used to help Rhino build the patch surface by first finding the best fit plane (PlaneThroughPt) through the selected and sampled points along curves. Then the surface deforms to match the points and sampled points. The Stiffness setting tells how much you allow the best fit plane to deform. The bigger the number, the "stiffer" and more rectangular and planar the resulting surface will be.

Preview to check the result.

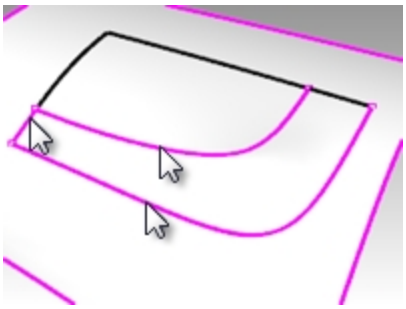
Adjust tangency to patch to the tangent direction of surfaces if the input curves are edges of existing surfaces.



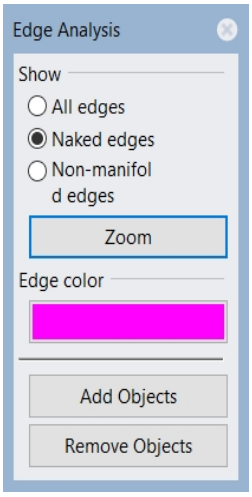
6. **Join** the surfaces.



7. Use the **ShowEdges** command *(Analyze menu: Edge tools > Show Edges)* to display naked edges.



If there are naked edges between the new patch surface and the existing polysurface the settings may need to be refined.



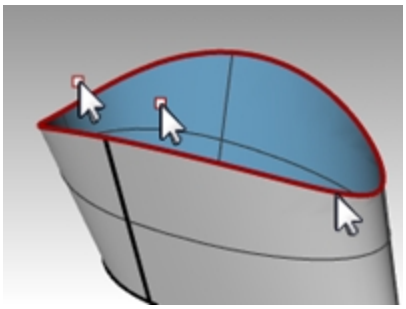
8. Check the results visually with **Zebra** analysis.

Patch options

The **Patch** command can use point objects as well as curves and surface edges as input. This exercise will use point and edge inputs to demonstrate how the Stiffness setting works.

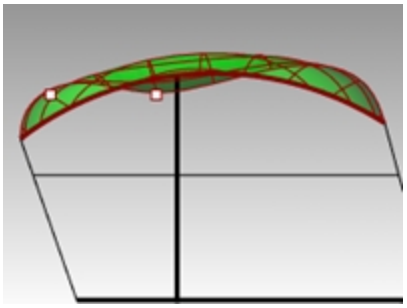
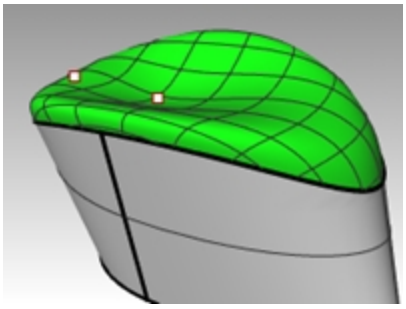
Exercise 6-3 Make a patch from an edge and points

1. Open the model **Patch Options.3dm**.
2. Start the **Patch** command *(Surface menu: Patch)* and select the two point objects and the top edge of the surface as input.
3. Check the **Adjust tangency** and **Automatic trim** options, set the **Surface spans** to **10** in each direction.
4. To get a good view of the two point objects, make the **Front** viewport the active viewport and set it to a wireframe or ghosted view.
5. Set the **Stiffness** to **0.1** and click the Preview button.

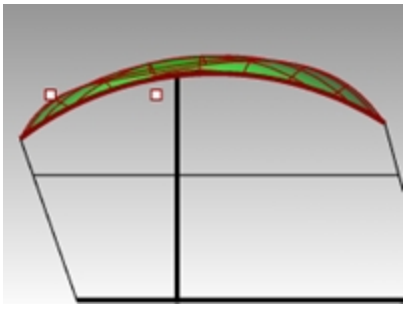
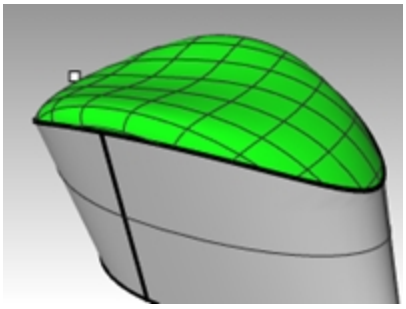


With lower setting for stiffness, the surface fits through the points while maintaining tangency at the surface edge. This can show abrupt changes or wrinkles in the surface.

6. Set the **Stiffness** to **5** and click the Preview button again.
   With higher stiffness settings, the patch surface is made stiffer and it may not pass through the input geometry. On the other hand, the surface is less apt to show abrupt changes or wrinkles, often making a smoother and better surface.
   With very high stiffness numbers, the edges also may have a tendency to pull away from the intended input edges.

High stiffness number = the more rectangular and planar the resulting surface will be
Low stiffness number = the smoother (fairer) the resulting surface will be
More spans = greater density of control points

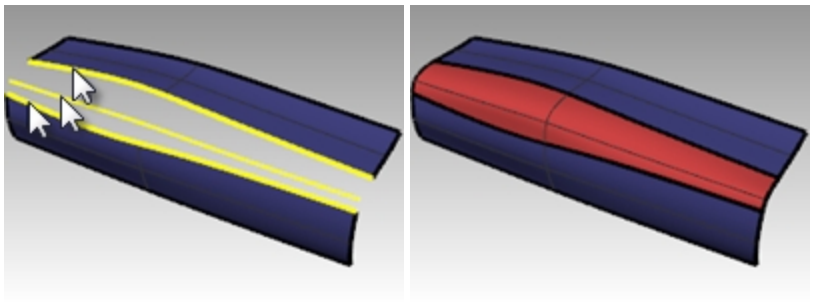



Lofting

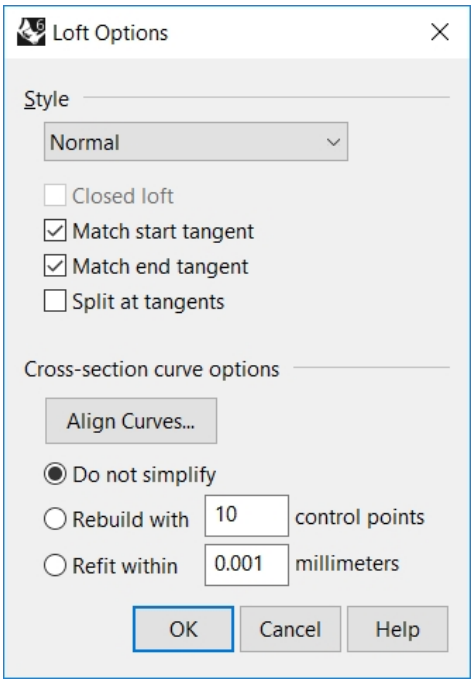
The **Loft** command also has built in options for surface continuity.

Exercise 6-4 Make a lofted surface

1. Open the model **Loft.3dm.**
2. Start the **Loft** command *(Surface menu: Loft)*.
3. Select the lower edge curve, the curve, then the upper edge curved, and press Enter.
   When picking the curves, pick near the same end of each curve. This will insure that you do not get a twist in the surface.

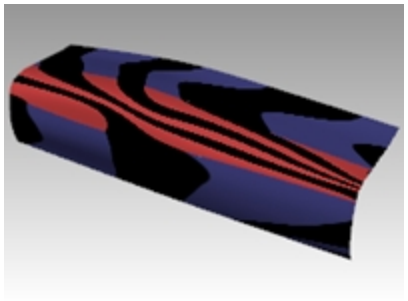


4. In the **Loft Options** dialog box, set the **Style** to **Normal**, and check the boxes for **Match start tangent** and **Match end tangent** options.
5. Press Enter when done.
   The new surface has G1 continuity to the original surfaces.



Style:
Loose—similar to control point curve
Straight—similar to polyline
Normal/Tight—similar to interpolated curve

6. Check the results with **Zebra** analysis.



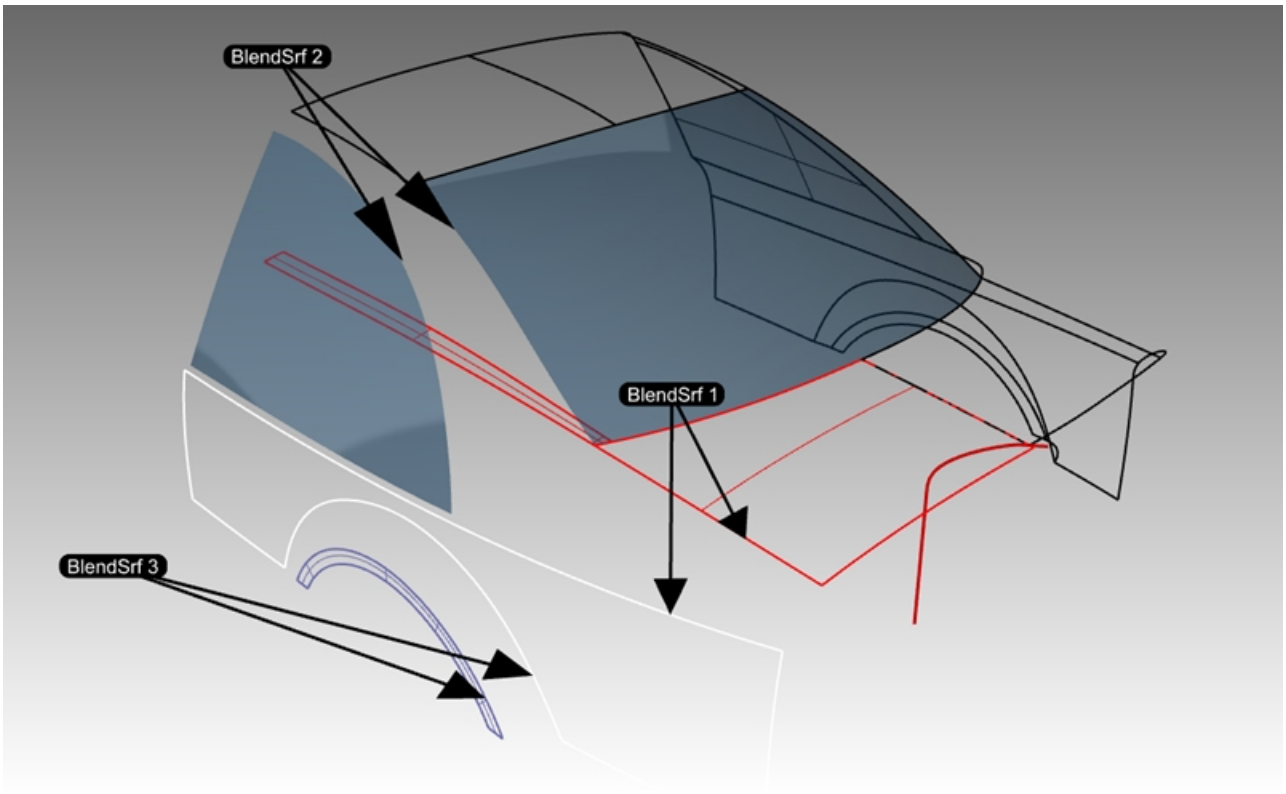
Blends

The next command that pays attention to continuity with adjoining surfaces is **BlendSrf**.

BlendSrf will also use the **Record History** setting.

If **Record History** is on in the Status Bar when the **BlendSrf** command is used to create a surface, editing the input curve will update the surface.

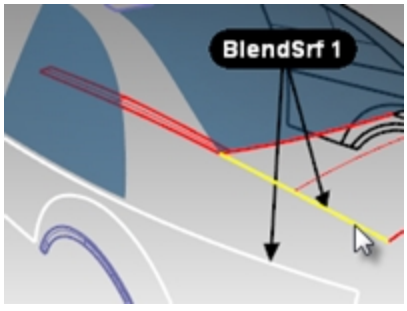
The three blends in this file will serve to illustrate the basic features of the **BlendSrf** command. The controls inside **BlendSrf** can be used to vary the character of the blended shape.



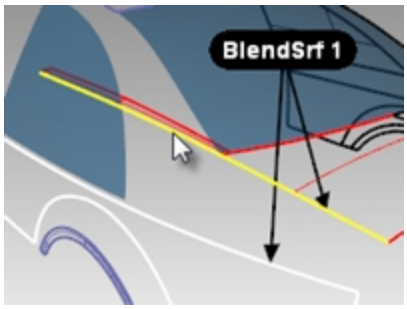
Exercise 6-5 Make a surface blend (BlendSrf 1)

In **BlendSrf 1**, we will create the transition between the trunk lid and side surfaces of the car body.

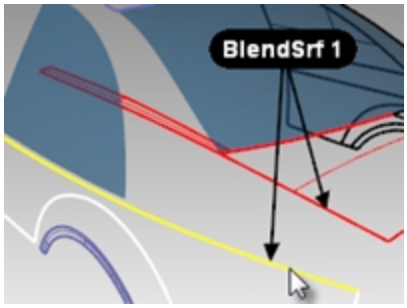
1. Open the model **Blend.3dm**.
2. Start the **BlendSrf** command *(Surface menu: Blend Surface)*.
3. You will blend the full length of the gap between the white side surface and the red surfaces. The ChainEdges option will allow you to select more than one edge segment.
   At the **Select first edge** prompt, click the command-line option **ChainEdges.**
4. In command-line, set options to **AutoChain=Yes** and **ChainContinuity=Tangency**.
5. Next select an edge along the red polysurface of the car body as marked in the file.



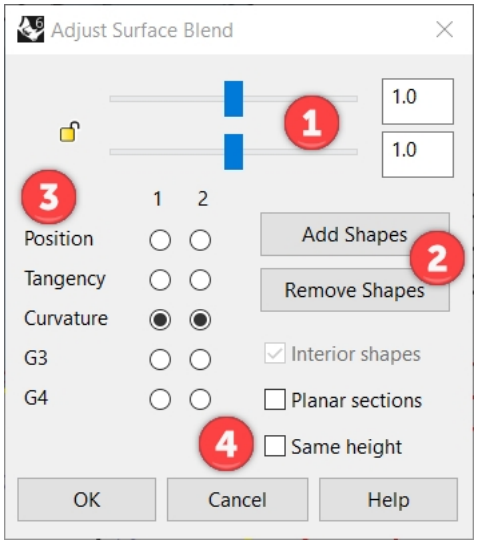
6. The edge from the adjacent surface is selected.



7. Press Enter to finish the selection for the first edge.
8. For the **Select segment for second edge**, select the top edge of the white side surface at the end of the edge nearest your initial pick for the first edge.
9. Press **Enter** to finish edge selection.
10. The **Adjust Surface Blend** dialog box with several controls appears. The next section will discuss these options.



Adjust Surface Blend dialog box



1. The sliders refer to the two default shape curves at the ends of the blend. Click the Lock icon to force both sides of the blend to be adjusted at the same time.
2. This button allows the user to add shape curves. These new shape curves have adjustable handles on them exactly like the default shape curves.
   Note: While it is sometimes useful to add shape curves, you should try to add as few as possible to achieve the shape you want. Interpolation among the shape curves is better if they are not too close together.
3. The radio buttons are for setting the continuity at each side of the blend; the edges are labeled 1 and 2 in the viewport.
4. There are check boxes for further options. These options will be discussed later in a separate exercise.

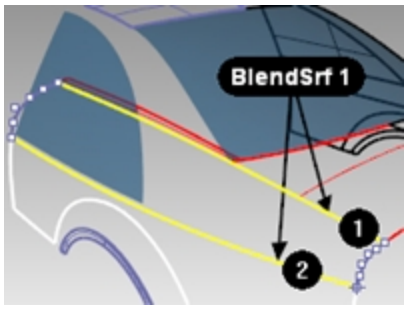
To configure the dialog for the Blend, do the following:

1. Clear the check boxes for **Same height** and **Planar sections** options.
2. Make sure the **Continuity** buttons are set to **Curvature**.
   A preview of the blend surface will be visible in the viewport.
   In the viewport, you will also notice a default pair of shape curves with points.
   These points are called *handles*.

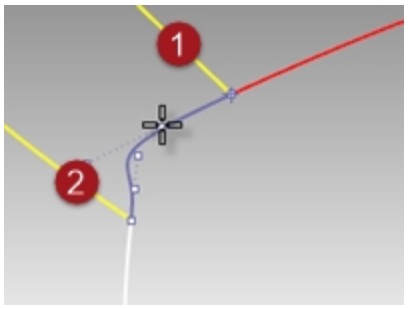
   The number of handles available on the shape curves varies according to the continuity settings in the dialog box.

   For example, if the continuity is set to **Curvature** for both shape curves 1 and 2, the curves will have six points

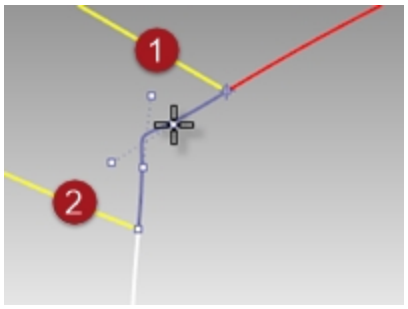
(three for each curve). If the continuity is set to **Tangent** for both shape curves 1 and 2, the curves will have four points (two for each curve).



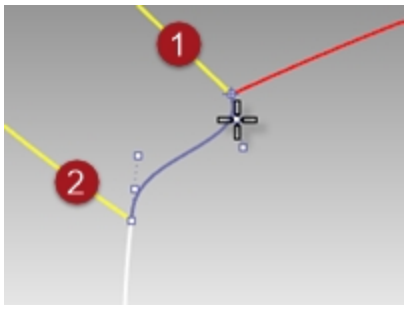
3. Try adjusting the handles on the shape curves. For example, at the rear of the car, make the blend sharper by moving the handles out so that they are crowded near the apex of the shape curve.
   The handles can be adjusted interactively on each shape curve to change the shape of the blend.
   Moving the handles changes the shape on one side of one shape curve.



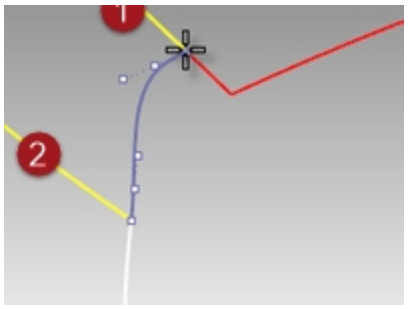
Press Shift while moving the handles to force both ends of the shape curve to be adjusted together. This is useful in maintaining symmetry on the blend shape.



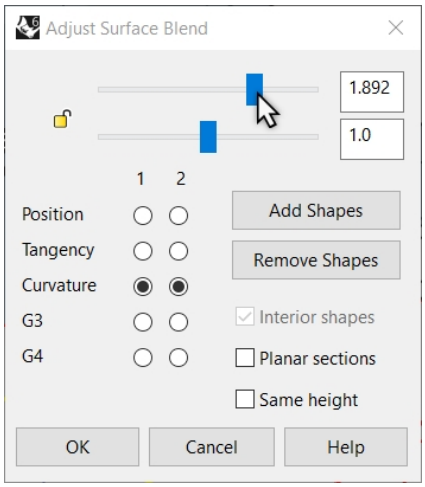
Press Alt while adjusting handles to rotate the handles and thus the direction of the shape curve relative to the edge.

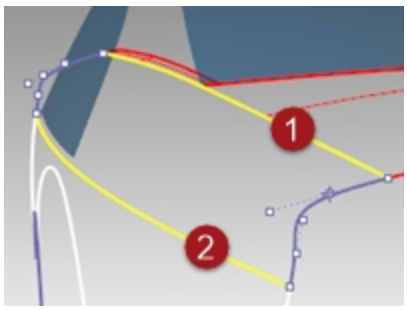


Move the handle at an end of a shape curve to change the location of the shape curve.

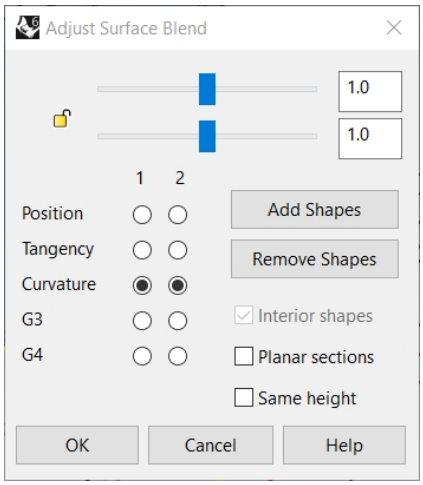


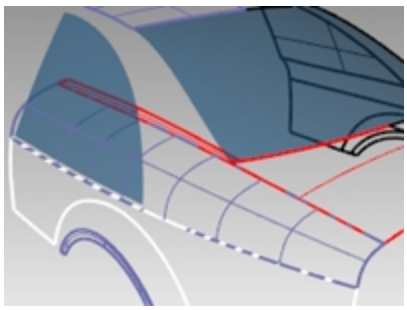
Use the sliders in the dialog box to change all shape curves together.
The top slider modifies all the shape curves near the original edge #1. The lower slider modifies all the shape curves near the original edge #2.





4. Adjust the settings in the dialog box to the default value of **1.0**, and click OK to make the blend surface.

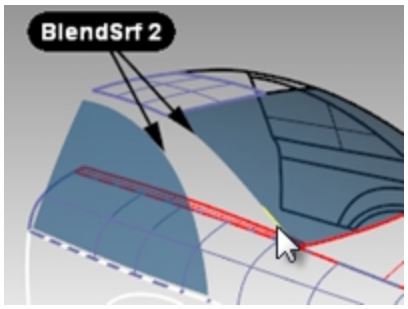




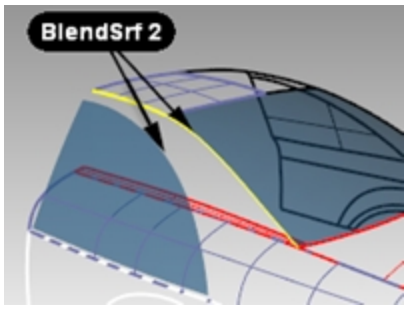
Make a surface blend with All (BlendSrf 2)

Next, we will blend between the roof rail and the side window.

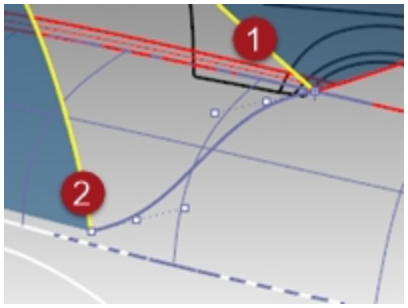
1. Start the **BlendSrf** command *(Surface menu: Blend Surface)*.
2. In the command-line, click the **ChainEdge** option.
3. In command-line, set options to **AutoChain=No**.
4. At the **Select segment for first edge** prompt, select an edge along the rear window of the car as marked in the file. Notice that only a small part of the edge is selected. Although the window is a single surface, the edges are split.

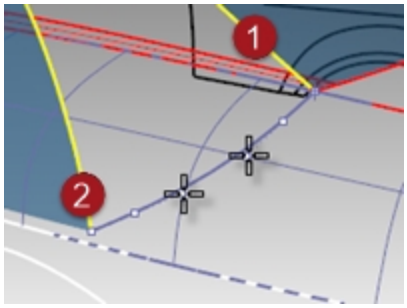


5. Next click the **All** option on the command-line to force chaining the edge fragments together.
   Notice that the edge of the roof panel is also added, since the edges are contiguous and tangent to one another.



6. Press Enter to finish the first edge selection.
7. For the **Select segment for second edge**, select the edge at the top of the side window.
8. Press **Enter** to finish edge selection.
9. The **Adjust Surface Blend** dialog box with several controls appears.
   Notice the default shape curve shows an S-shape at the lower end of the blend area.
   Press the Alt key while dragging the handles to align the blend curve in a more natural way.



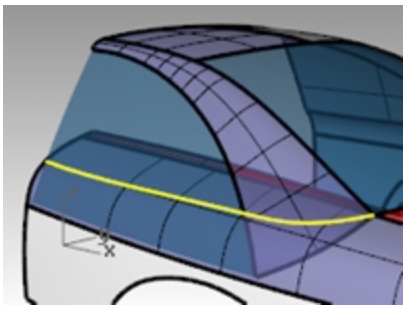


10. Click OK to make the blend surface.

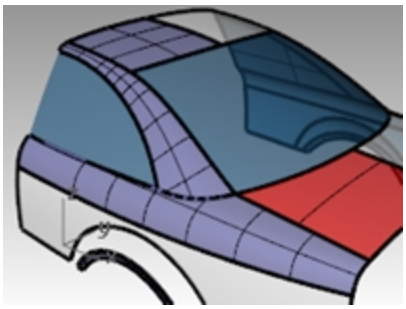
Clean up the excess surfaces

To clean up the excess surfaces, we will trim the surfaces to each other. Since the intersections of the surfaces do not go all the way to the edges of all of the surfaces, we can make intersection curves, join them, and then extend them as needed on the surfaces.

1. Start the **IntersectTwoSets** command *(Curve menu: Curve from Objects > Intersection of Two Sets)*.
2. As the first set of objects to intersect, select the side window and the roof blend surface you just completed. Press Enter.
3. As the second set of objects to intersect with the first set, select the first blend surface you made. Press Enter.
4. Join the resulting curves.
5. Select the joined curve.
6. Start **ExtendCrvOnSrf** *(Curve menu: Extend Curve > Curve on Surface)* and choose the lower blend surface as the surface to extend on.



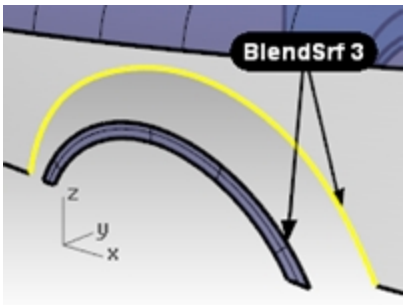
7. Trim the bottom of the side window, the lower end of the roof rail blend, and the side blend inside the roof and glass area.

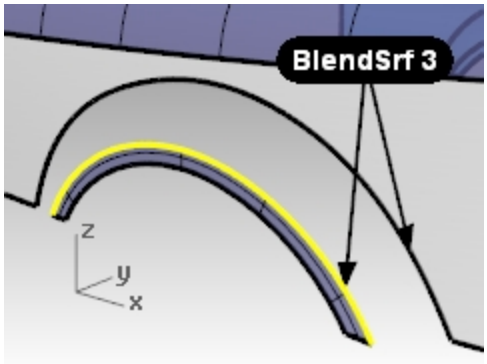


Make a surface blend (BlendSrf 3)

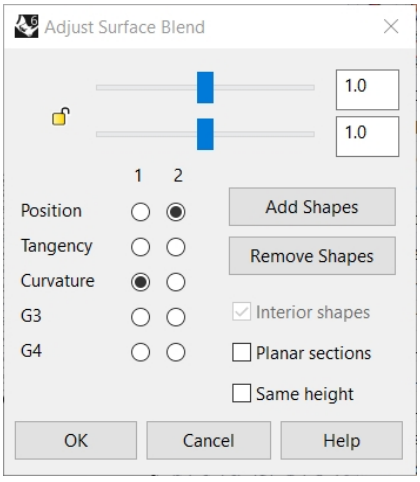
Lastly, we will blend between the wheel arch and the side of the car.

1. Start the **BlendSrf** command *(Surface menu: Blend Surface)*.
2. For the **Segment for first edge**, select an edge of the wheel arch on the side of the car and Press Enter.
3. For the **Segment for second edge**, select the other edge of the wheel arch.

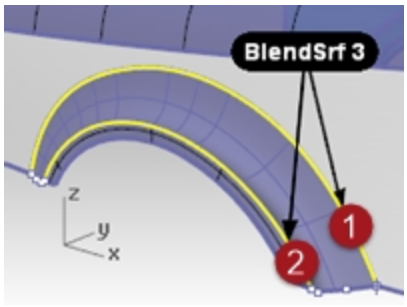




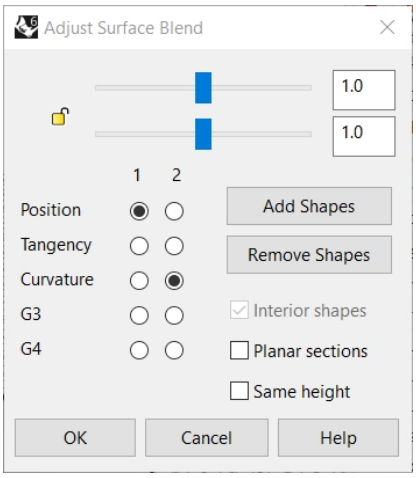
4. Change the continuity settings in the dialog box so that one edge has **Position** continuity (G0) and the other has **Curvature** continuity (G2), and check the Preview option.



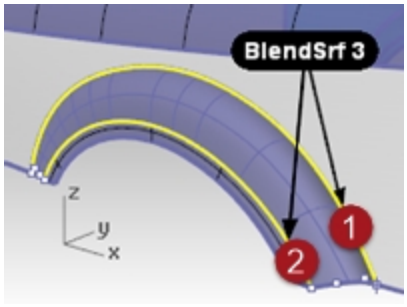
This will allow you to have a hard edge at one of the edges.



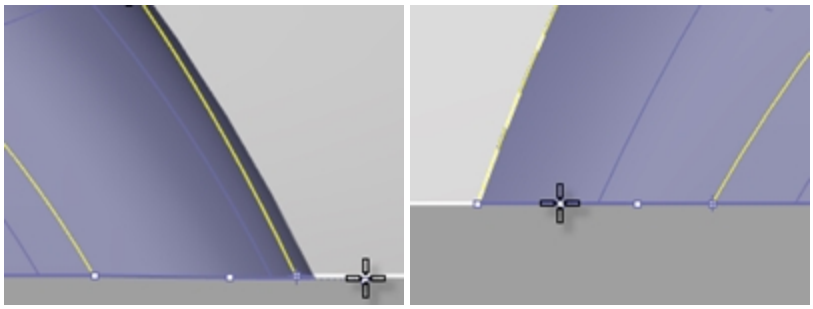
5. Switch the continuity settings to the opposite edges to change the character of the blend.



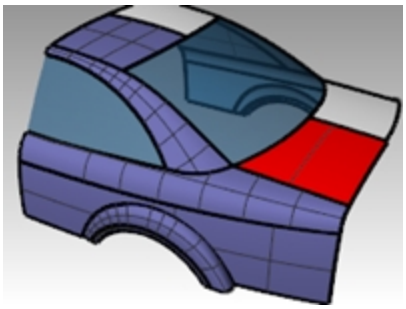
You will probably have to rotate the shape curves at both sides of the wheel arch so that they align with the bottom edge of the side.
In this case, it is easier to do this in the Front viewport.



6. Press the Alt key while dragging the handles to align the blend curve to the bottom edge of the side.

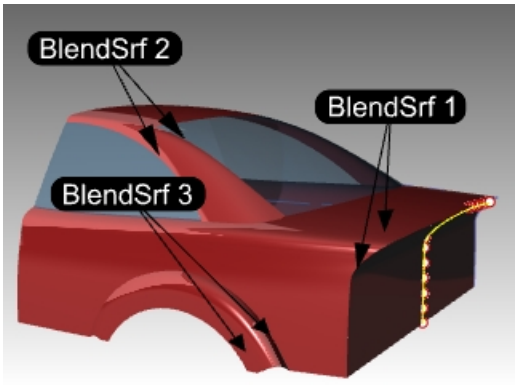


7. Click OK in the dialog box to make the blend surface.



On your own

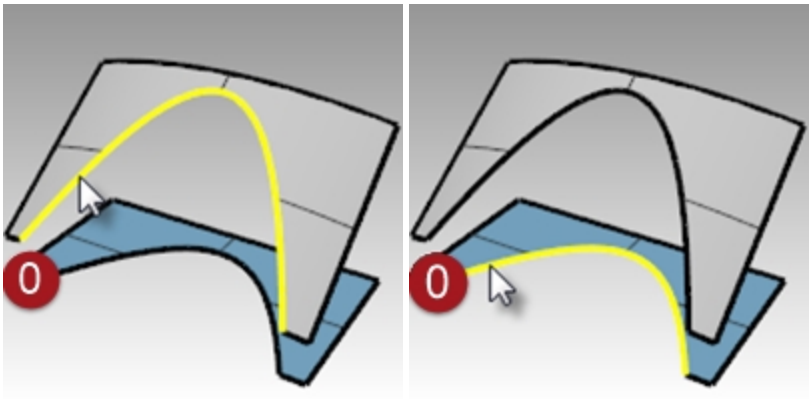
Extrude the highlighted curve to create the back surface of the car. Use the same methods you used above to find the intersection curve, extend to the surface and trim.



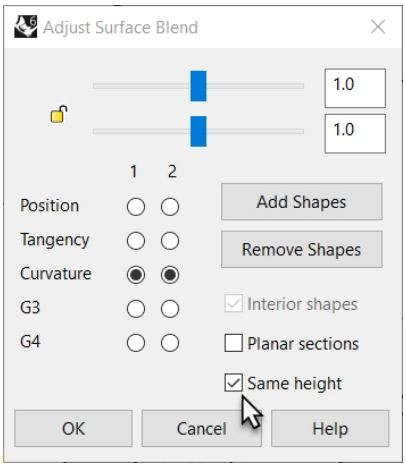
Use Blend options

In the following exercise, we will first make a surface blend that creates a self-intersecting surface. Then we will use the surface blend options to correct the problem.

1. Open the model **BlendSrf Options.3dm**.
2. Start the **BlendSrf** command *(Surface menu: Blend Surface)* and select the deeply curved edges of the pair of surfaces marked 0.

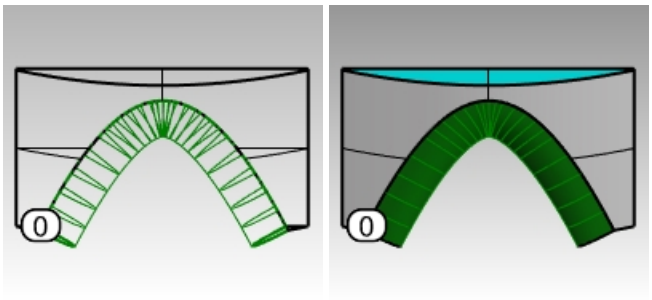


3. In the dialog box, check the **Same height** option, and the bulge sliders are set to **1.0**.
4. Click OK.



5. Zoom in on the surface you just created in the Top viewport.

   Look closely at the middle of the blend surface in this view using a wireframe viewport. Notice the blend has forced the surface to be self-intersecting in the middle. The isocurves cross each other and make a pinch or crease here.



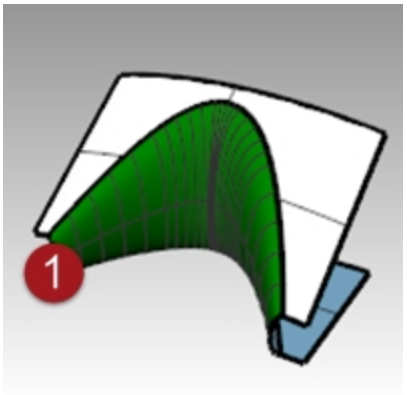
Surface blend options

To avoid self-intersecting or pinched surfaces when creating a blend you can Adjust Blend Bulge sliders, use Same height shapes, or use the Planar sections option.

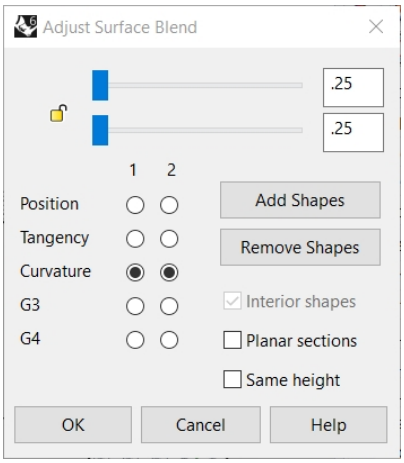
In the following examples, we will look at each of these options.

Exercise 6-6 Make a surface blend with options

1. Start the **BlendSrf** command and select the edges of the pair of surfaces marked (1).
2. Adjust the sliders to make the bulge of the surface less than 1.

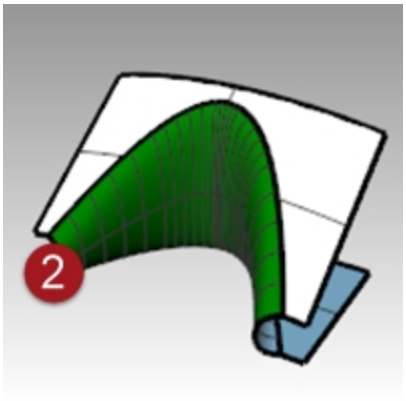


   A number between 0.2 and 0.3 seems to work best.

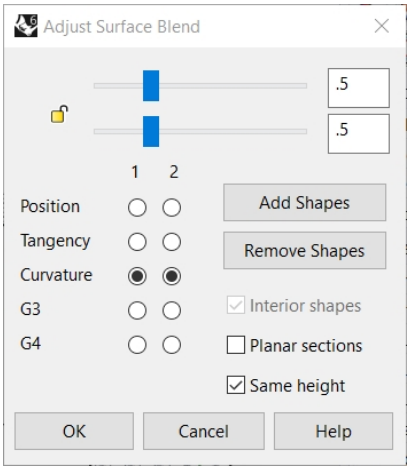


The profiles of the cross sections at each end of the blend as well as any you may add between will update to preview the bulge. Notice that the surface is not pinched in the middle.

3. Start the **BlendSrf** command and select the edges of the pair of surfaces marked 2.

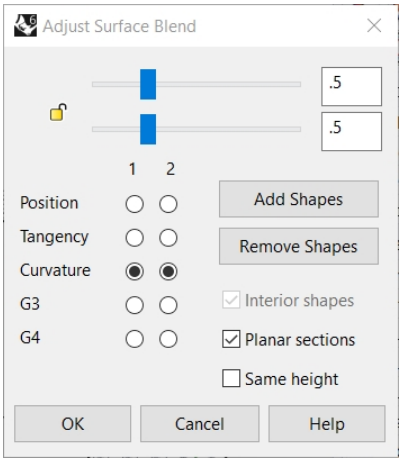


4. Change the **Bulge** to **0.5**, but check the **Same height shapes** options.
   The **Same height shapes** option overrides the tendency of the blend surface to get fatter or deeper according to how far apart the edges are. The height will be the same in the center as it is at each end. This also has the effect of making the sections of the blend push out less and therefore not cross each other out in the middle area.

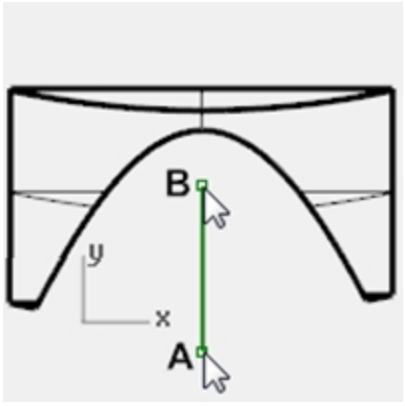


5. Start the **BlendSrf** command and select the edges of the pair of surfaces marked **3**.

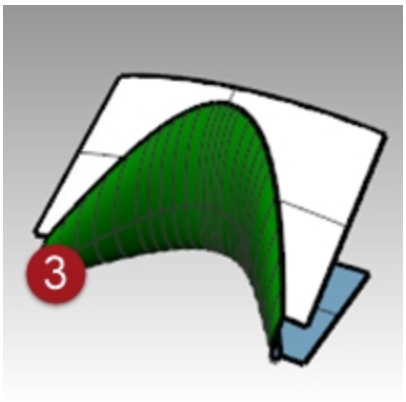
6. Pick the edges in the usual way.
   Use the same bulge settings as the last pair of surfaces.
7. In the dialog box, check the **Planar sections** option and clear the **Same height** option.
   You are now asked to define which plane the sections of the surface should be parallel to. This is defined by clicking two points in any viewport.



8. Click once anywhere in the **Top** viewport at **A** then with Ortho on, click again in the **Top** viewport **B** in the direction of the y-axis.



The resulting surface (3) has its isocurves arranged parallel to the plane defined in the Planar sections portion of the command. The isocurves do not intersect in the middle of the surface since they are parallel the y-axis.



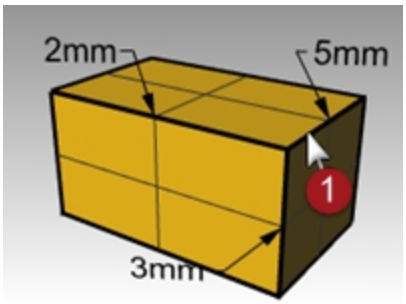
Fillets, blends, and corners

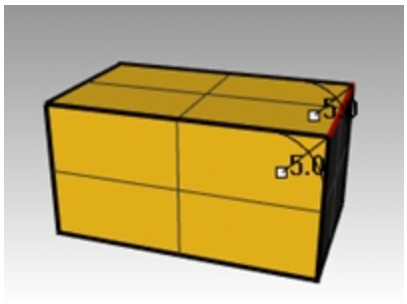
In this exercise, you will see a variety of ways to make transition surfaces with FilletEdge, BlendEdge, ChamferEdge and Patch commands.

While Rhino has automated functions for making fillets, several situations take manual techniques. In this section, we will discuss making corners with different fillet radii, variable radius fillets and blends, and fillet transitions.

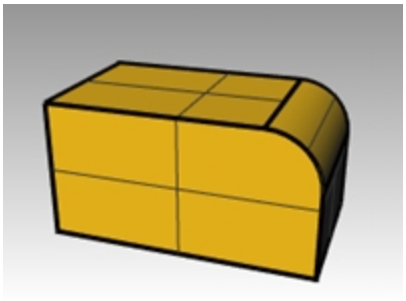
Exercise 6-7 Make a corner fillet with three different radii

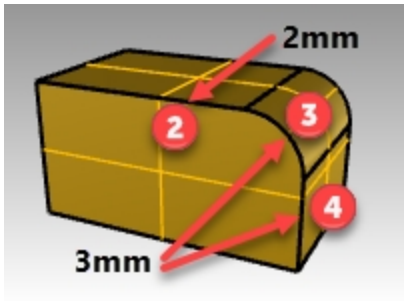
1. Open the model **Corner Fillet.3dm**.
2. Use the **FilletEdge** command *(Solid menu: Fillet Edge > Fillet Edge)* to fillet edge (1) with a radius of **5mm**.

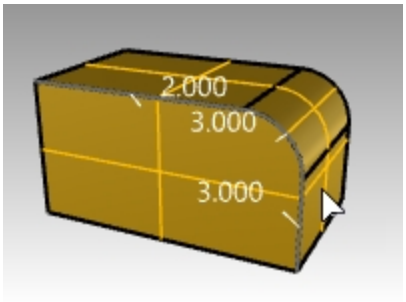




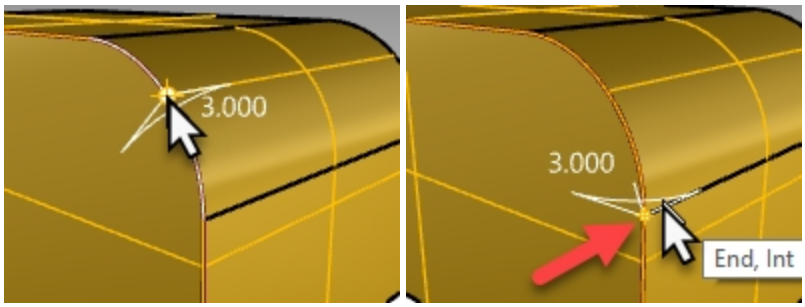
3. Use the **FilletEdge** command *(Solid menu: Fillet Edge > Fillet Edge)* to fillet edge (2) with a radius of 2mm, and next two edges (3) and (4) with a radius of 3mm. Enter
   Note: Change the value for **NextRadius** to 3 before selecting the second edge.



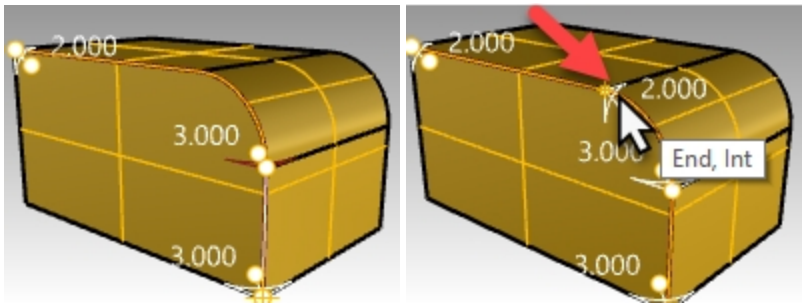




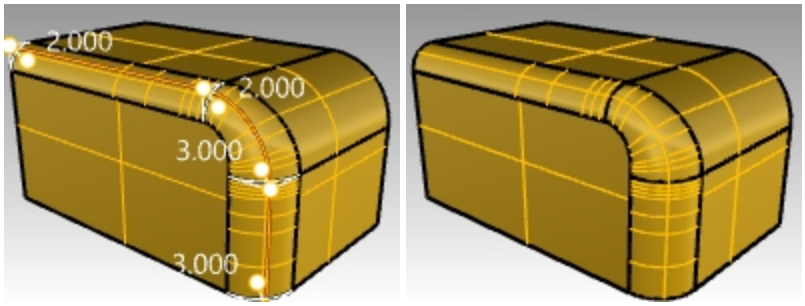
4. Using osnaps, drag the **3.0** radius handle to the end of the arc or vertical edge.



5. Use the **AddHandle** option. Set the **Current Radius** to **2.0** , the pick the new handle location at the other end of the arc, and Enter.



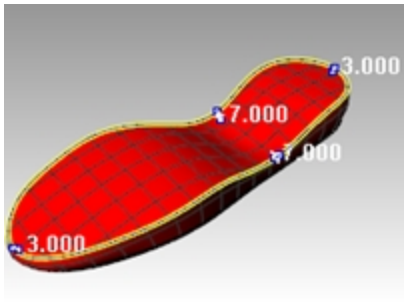
6. Use **Preview=Yes** to see the results of the handle edits, and then press Enter to complete the fillet.



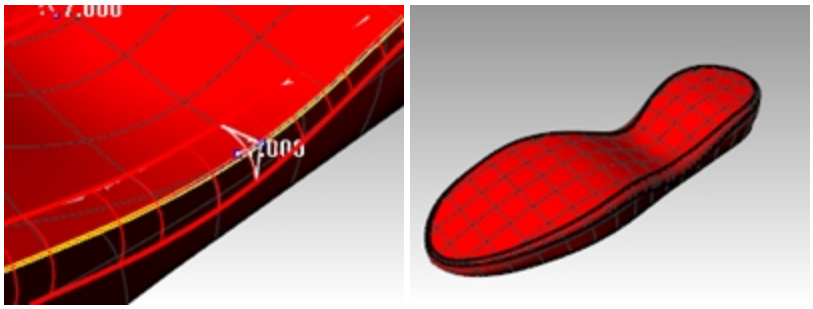
Now the fillet radius will varry from 2mm to 3mm on the arc edge.

Exercise 6-8 Make a variable radius blend

1. Open the model **Sandal Sole.3dm**.
2. Use the **BlendEdge** command *(Solid menu: Fillet Edge > Blend Edge)* to make a variable radius blend on the bottom of the sole. Start with a radius of 3mm.
3. Use the **AddHandle** option to add additional radii around the bottom of the sole.
4. Add another **3mm** radius to the front of the sole and then add a **7mm** radius to the instep on both sides.

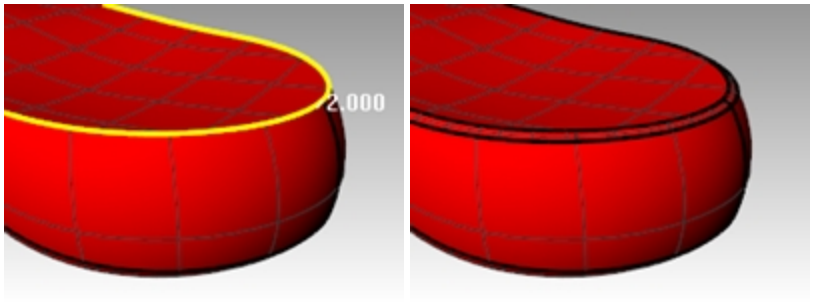


5. Preview the blend and adjust the handles as needed, and then press Enter to make the blend.



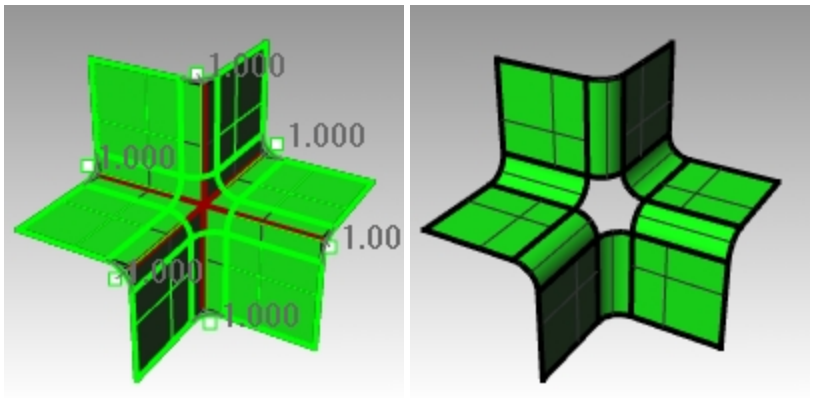
Make an edge chamfer

1. Use the **ChamferEdge** command *(Solid menu: Fillet Edge > Chamfer Edge)* to make a 2mm chamfer around the top edge of the sole.
   This command, like the **FilletEdge** command and the **BlendEdge** command, allow for the addition of handles with different values to create a variable distance chamfer.
2. Preview the chamfer and make adjustments as needed, then press Enter to make the chamfer.

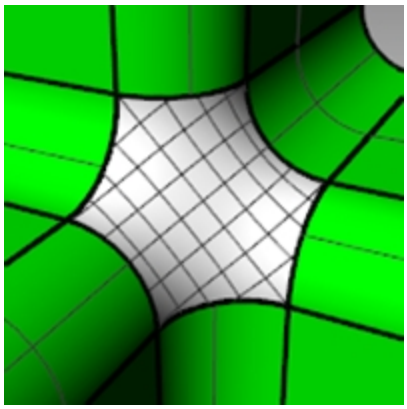


Exercise 6-9 Make a six-way fillet using a patch

1. Open the model **Fillet Edge.3dm**.
2. Use the **FilletEdge** command *(Solid menu: Fillet Edge > Fillet Edge)*, with **Radius=1**, to fillet all the joined edges at the same time.

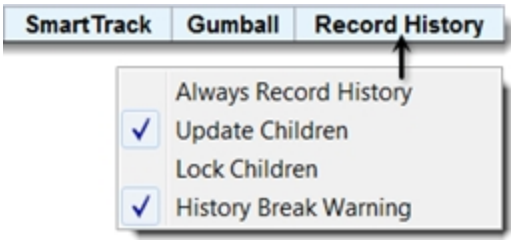


3. Use the **Patch** command *(Surface menu: Patch)* to fill in the opening at the center.
4. Select all six edges to define the patch.
5. In the **Patch Options** dialog box, check the **Adjust Tangency** and **Automatic Trim** options.
6. Change the **Surface U** and **V Spans** to **10**, and the **Stiffness** to **2**.
   Note: When the area to fill has more than four edges, the **Patch** command works better than the **NetworkSrf** command.



Chapter 7 - Modeling with history

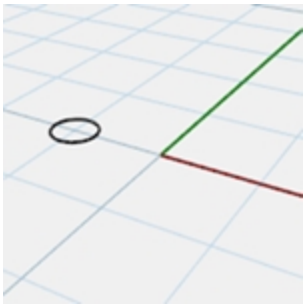
History allows editing or updating objects by editing the input geometry that was used to create the objects. History is useful when there is a need to edit the input of a command or when transformed copies of an object need to stay matched to the original. Only certain commands support history.



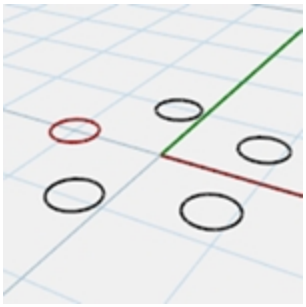
History is not the same as a “feature” or “parametric.” History information is saved in the Rhino *.3dm file.

A simple example

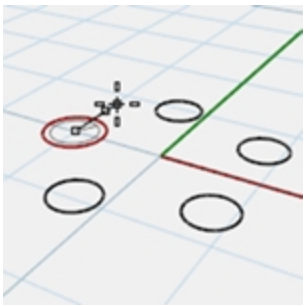
1. Draw a circle.



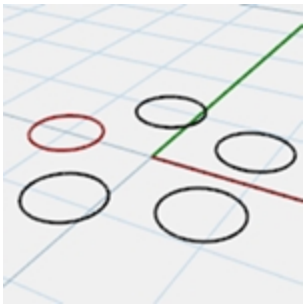
2. Turn on Record History and Array the circle.



3. Scale the original.

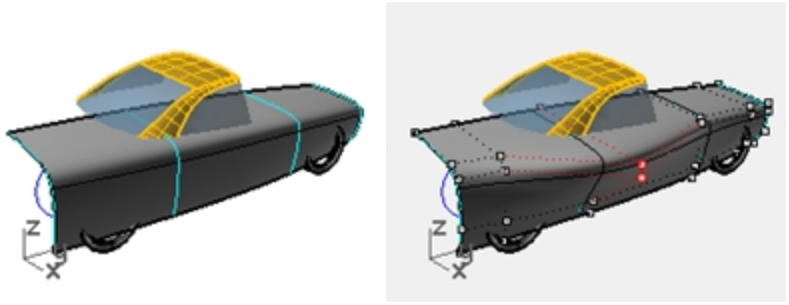


4. Watch the arrayed circles update.



Make a lofted surface

1. Open the model **History_Intro.3dm**.
2. Select the four cyan colored curves.
3. Start the **Loft** command *(Surface menu: Loft)*, select **Normal** style, and click OK.
   Loft the curves to generate a smooth surface.



4. Turn on the control points and edit the surface.
   Turning on surface control points allows the surface to be edited directly as always. However editing the input curves does nothing to change the surface.
5. **Undo** or delete the loft.

Activating history

History recording is off by default. It must be turned on before running a command to record history for that command. The status of history recording is indicated on the Record History pane on the status bar. If the text in this pane is bold, recording is active. Click the pane to change the status.

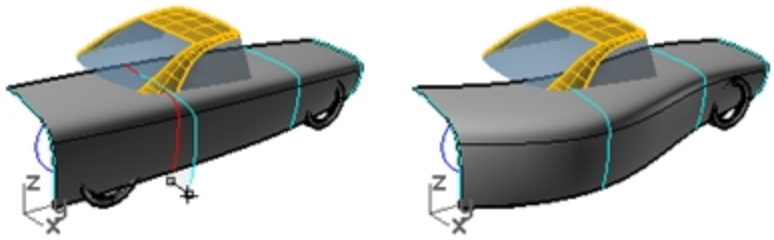
To record history for a particular command, click the Record History pane, and then start a command that pays attention to history.

Why is history off by default?

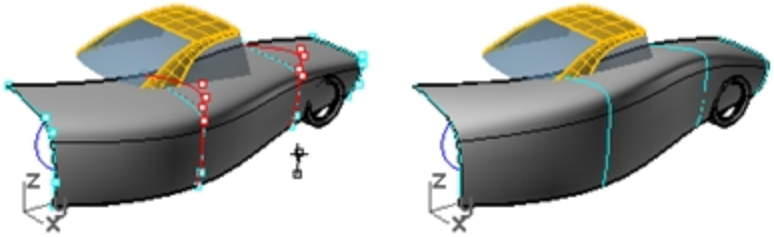
- Unpredictable results. For example, if you Copy with history and make a change, all of the children start to update.
- File size is larger with history enabled.

Make a lofted surface with history

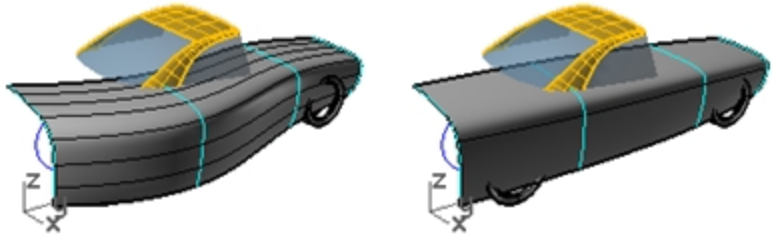
1. Click in the **Record History** pane in the status bar to make it bold and active.
2. Select the four cyan colored curves.
3. Start the **Loft** command *(Surface menu: Loft)*, select **Normal** style, click OK.
   Notice the **Record History** pane is no longer bold once a command has run.



4. Select one of the input curves and move it.
   The loft surface updates to reflect this new position.
5. Turn on control points of the input curves.
6. Edit the points and the surface will update.



7. Select the curves and **Rebuild** *(Edit menu: Rebuild)* them to **10** points.
   The lofted surface updates to reflect this change as well. Changing degree of the parent curves will also change the degree of the child surface in that direction.
8. **Undo** the three previous steps.



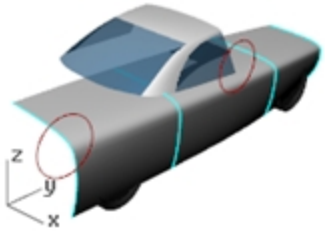
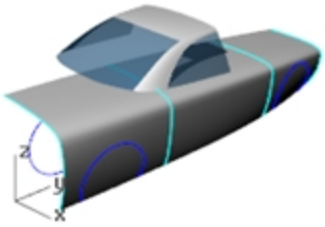
Steps in the history chain

- The command must support History.
- History recording must be active when the command is actually run. By default, History recording is turned off and must be activated each time a command is run for which the user wants to record history.
- History updating must be on. This is on by default. When it is on, edits to input objects are immediately reflected in the updated output.
- History can be nested; for example, a curve can be projected onto a lofted surface, and the curve will follow the changes to the lofted surface.

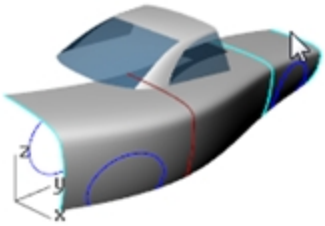
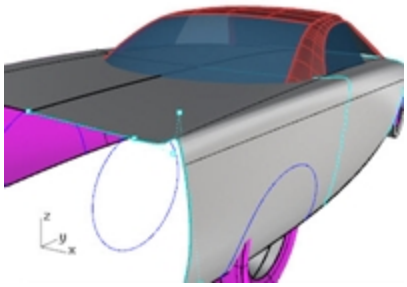
Project a curve onto a surface with history

This part of the exercise will demonstrate an example of nested history. We will be projecting the wheel cutout curves onto the lofted surface.

1. First, we will change the construction plane for the **Perspective** viewport.
2. On the **CPlanes** toolbar tab, click **Set CPlane World Right**.
3. Click in the **Record History** pane in the status bar to make it active.
4. Use the **Project** command *(Curve menu: Curve from Object > Project)* to project the two wheel cutout curves onto the lofted surface.

5. Set the **CPlane back to World Top** *(CPlanes toolbar tab, Set CPlane World Top)*.
6. Select one of the input curves for the loft. Modify it by:
   Moving or scaling
   Control point editing
   Hint: Gumball can be helpful here.
   The projected wheel cutout curves update to follow the surface.

Note: Any editing of the outputs will 'break' History and the connection between inputs and outputs will be lost. Rhino will put up a warning box when this happens and the user can either Undo to restore the connection, or continue editing and accept the break in History.

History-enabled commands

A list of the history-enabled commands is maintained in the **History** command help topic.

History-enabled commands

History
HistoryPurge
SelObjectsWithHistory
SelChildren
SelParents

History toolbar

History Options

Inputs to a History-enabled command are called Parents in Rhino and the outputs are called Children.

Right-click the Record History pane to change the following options:

Always Record History

This option changes the default behavior so any eligible command will always record history. Use this option with caution. In addition to unnecessarily increasing the file size, it can lead to unexpected behavior. To clear history on particular objects or on all objects, use the HistoryPurge command.

Update Children

Causes child objects to update each time the parent object changes. This increases the time it takes to update complex objects. For very complex edits on the parent objects, turn off updating, make the changes, and then turn Update Children on so that the update happens only once.

Lock Children

This option sets child objects to a locked state. Since directly editing the child objects breaks the connection to the parent objects, locking the child objects prevents accidental editing. In addition, selecting child objects can be cumbersome if they are in the same location as the parent objects. Locked child objects still update when the parent objects are edited.

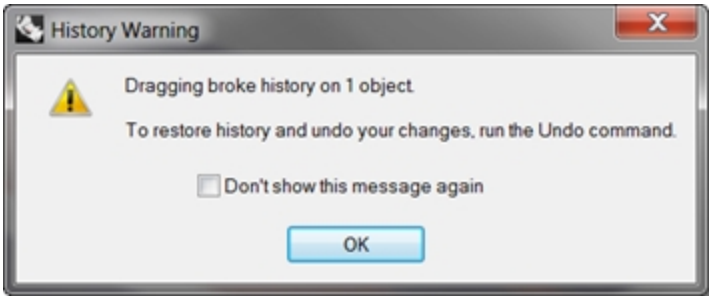
History Break Warning

This option displays a warning if an operation breaks the connection of a child object to its parent objects. The Undo command will restore history.

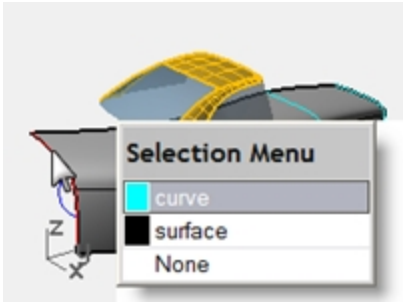
Use the History command to control history recording, updating, locking, and warnings.

Change history options

1. Click on a curve to get the multi-select box.
   If you edit the surface in any way, History for the object will break and Rhino will warn you about this.
2. Select the surface and drag it. Rhino will warn you that dragging broke history.
3. Click OK.



Make sure to Undo after getting a "broken History" warning to restore the connection between inputs and output.



4. Right-click in the **Record History** pane and check the **Lock Children** option.
   This will make it impossible to edit a child in any way that will break history, but you can select it to change its object properties or layer, etc.

Advanced Surfacing Strategies

There are several ways to approach making a soft box shape like the illustration below.

In this exercise, we explore two different methods to make the surfaces using the same underlying curves. Our design curves in this example are all tangent arcs.

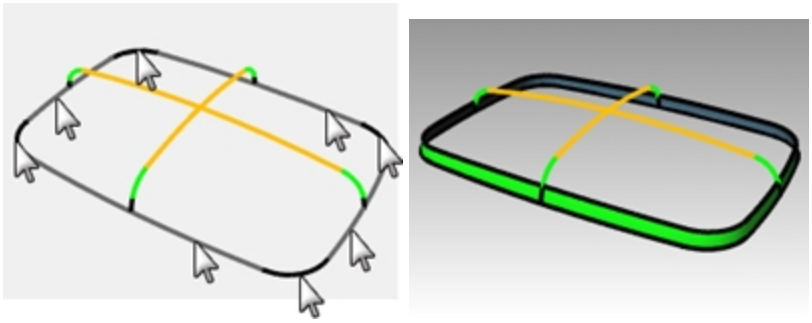
The first method will use the curves directly. For second method, we will plan ahead and try to take into account the simple underlying shapes suggested by the design curves.

The two approaches are different, but one is not inherently better or contradictory to the other.

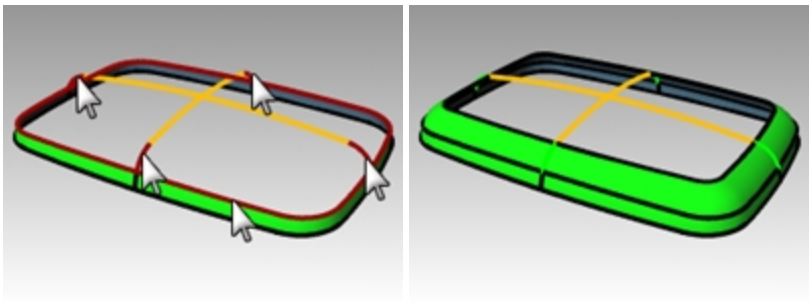
Exercise 7-1 Soft corners (part 1)

Make a rectangular shape with a curved top and soft corners

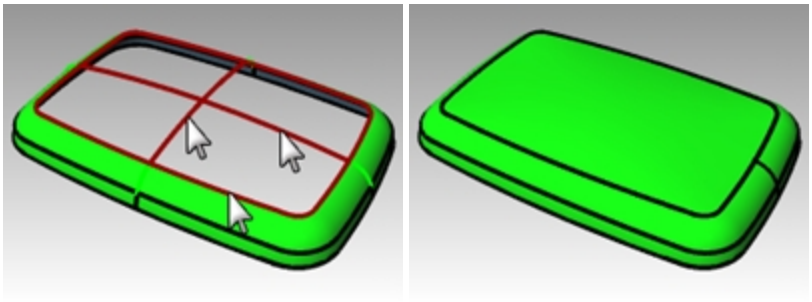
1. Open the model **Soft Corners.3dm**.
2. Use the **Join** command *(Edit menu: Join)* to join the arcs that form the base rectangular shape.
3. Change to the **03 Sweeps** layer.
4. Use the **Sweep1** command *(Surface menu: Sweep 1 Rail)* to make the first surface.
5. In the **Sweep 1 Rail Options** dialog box, check the **Closed sweep** option, then click OK.



6. Repeat the **Sweep1** command to make the second surface.
7. Pick the top edge of the surface you just created, then select the cross-sections in order, and press Enter.
8. In the **Sweep 1 Rail Options** dialog box, change the **Style** to **Align with surface**, and click OK.
   This will insure tangent continuity with the first surface.



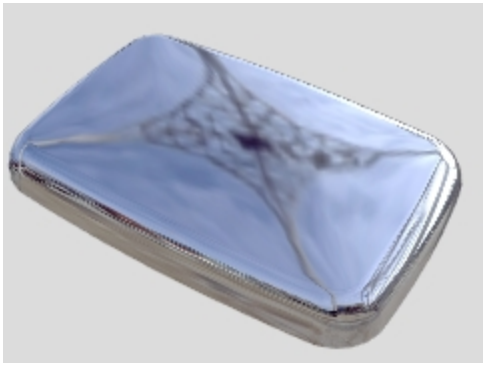
9. Use the **Patch** command *(Surface menu: Patch)* to fill in the opening at the center.



Check the surface with an environment map

1. Select the surfaces you made.
2. Start the **EMap** command *(Analyze menu: Surface > Environment map)*.
3. Click **Adjust mesh** and refine the mesh in a similar way that you did for the **Zebra** surface analysis command.
4. Choose the **Arches.png** or **Space Needle.png** from the drop-down list in the **EMap Options** dialog box.

5. Tumble the view around.
   Note the top surface has a pronounced X-shape in the pseudo-reflections. The surface is not a clean interpretation of the original input curves- there is this extra distortion across the top surface. So, even though we have hit all the input curves, our surfaces are not necessarily very nice.



Soft corners - another method

This time, we will look at the input curves and make some judgments about the best way to construct the surfaces.

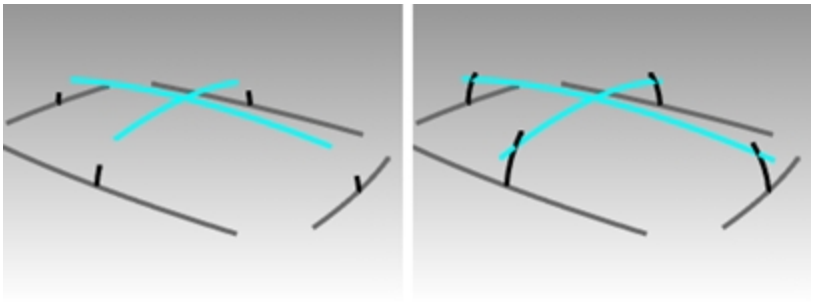
One thing to keep in mind is that it is important to make primary surfaces with good curvature characteristics before building the secondary, or transitional surfaces.

Primary surfaces will be the ones that define the overall shape. They tend to have relatively even curvature, and less curvature than the transitions. Transition surfaces, as the term implies, provide transitions between the primary shape surfaces. These surfaces tend to have higher curvature than the primary surfaces. Fillets and blends, for example, are generally added as transition surfaces.

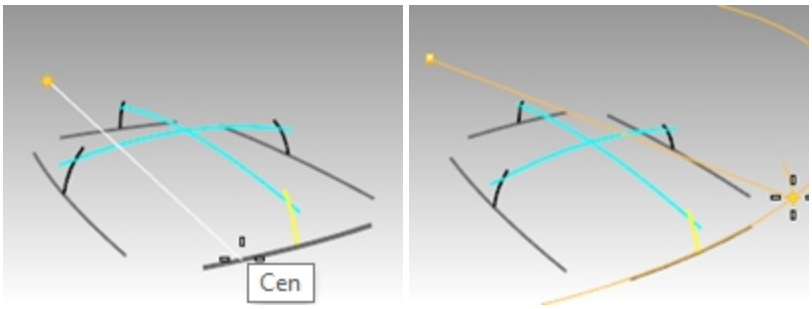
In this example, we have the four side surfaces and the top surface as primary surfaces. We will add the corners afterward as fillets. Since in our example the input curves are entirely made up of tangent arcs, we can define the side and top surfaces as revolved surfaces. This type of surface is very exact and simple.

Exercise 7-2 Soft corners (Part 2)

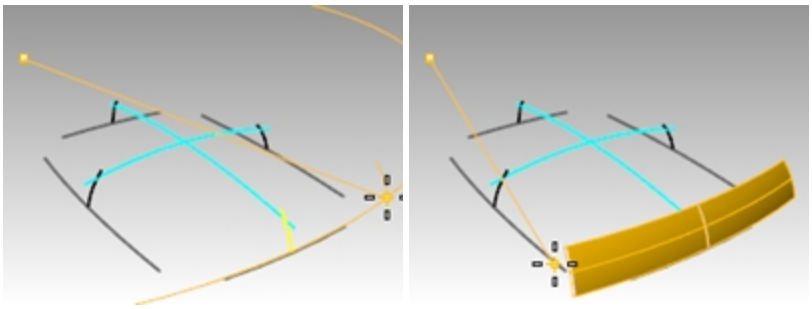
1. Change to the **02 Separate Curves** layer and turn all the other layers off.
2. Hide the fillet curve at each corner, and the green cross-section curves.
3. **Lock** the red curves.
4. Use the **Extend** command *(Curve menu: Extend Curve)* with an **ExtensionLength=10** to extend both ends of the cyan curves and the top of each black arc, press Enter to complete the command.
   Each arc is extended at each end using the existing arc radius.



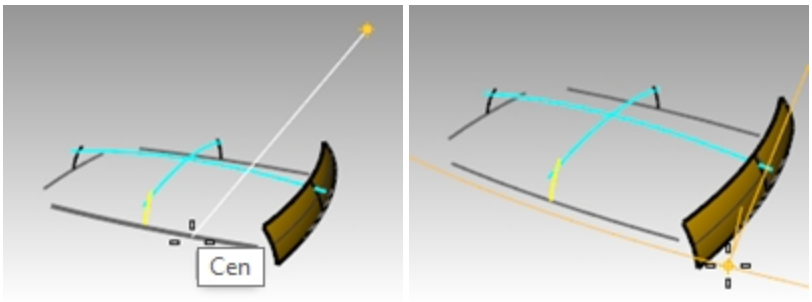
   The goal is to extend the arcs enough that they will intersect one another as in the illustration. The exact amount is not critical.
5. Change to the **04 Surfaces** layer.
6. Use the **Revolve** command *(Surface menu: Revolve)* to make surfaces from two adjacent extended vertical curves.
7. Snap to the center of the base curve to place the Start of revolve axis.
8. Press Enter to use **CPlane z-axis direction** for the **End of revolve axis**.
   If you are in a **Perspective viewport**, this option will automatically set the axis vertical and save the trouble of locating the second point.

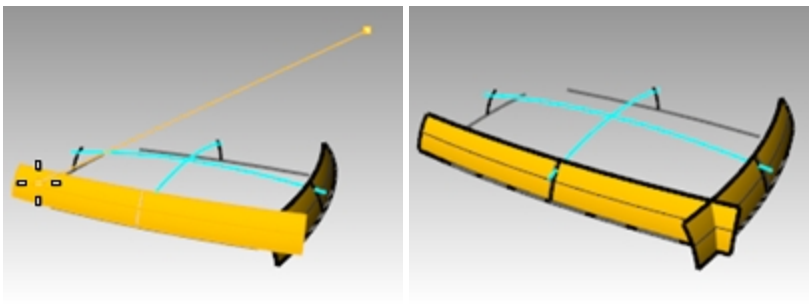


9. Pick the **Start angle** as shown in the image that is somewhere outside the span of the desired final surface. Make sure **Ortho** is **off** at this stage.
   The goal is to make a surface that is larger than will eventually be needed to make the box, so the exact starting and ending points are not critical.

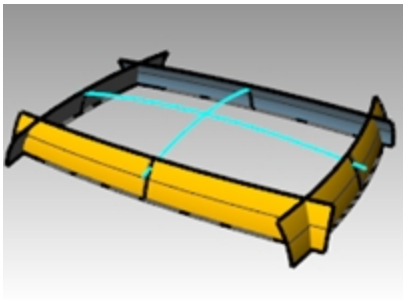


10. Pick another point for **Revolution angle** to create the vertical surface.
11. Create an adjacent vertical surface in the same way.





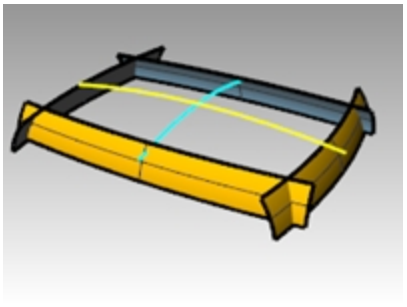
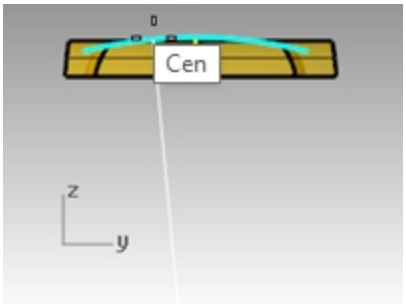
12. Use the **MX** and **MY** aliases you made on the first day to **Mirror** each of the surfaces around the origin.



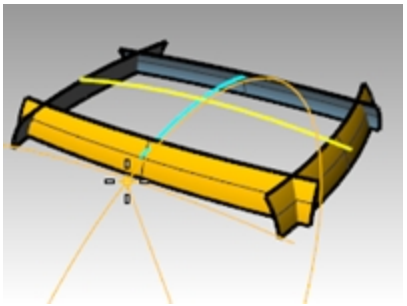
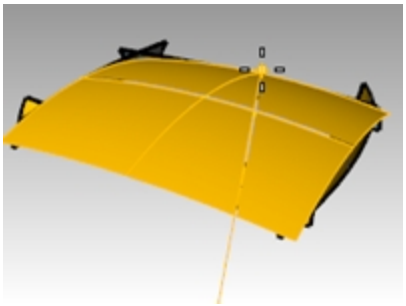
Make the top surface

In this example, we will make the top surface by revolving one of the top curves around the center of the other top curve. Since we will be working in the **Perspective** view, it will be necessary to change the construction plane for that viewport.

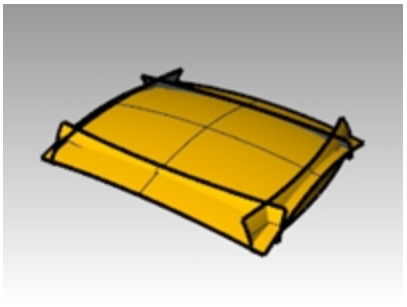
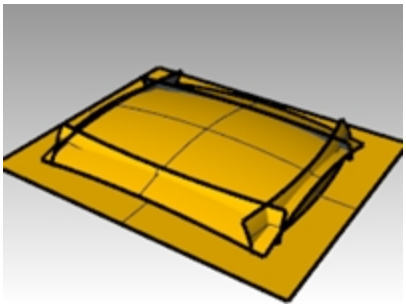
1. Use the **Revolve** command *(Surface menu: Revolve)* to make the top surface from the longer top arc.
2. Snap to the center of the shorter top arc in the **Right** viewport to place the **Start of revolve axis**.

3. Press Enter to use **CPlane z-axis** direction for the **End of revolve axis**.
4. Pick the **Start angle** as shown.
5. Pick another point for **Revolution angle** to create the top surface.

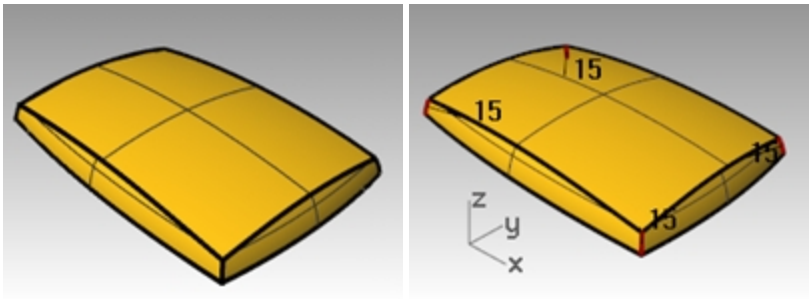



6. Use the **CutPlane** command *(Surface menu: Plane > Cutting Plane)* to make a cutting plane at the origin in the z-axis.

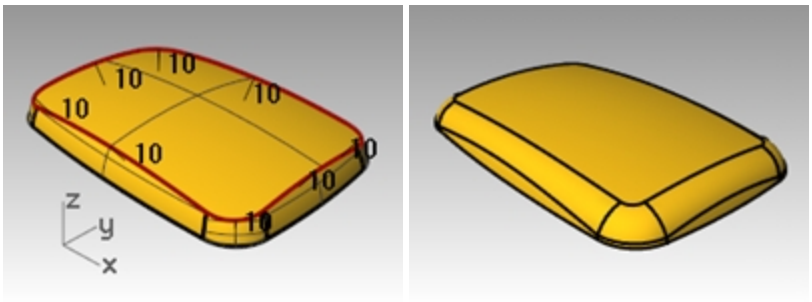



Make the surfaces into a solid

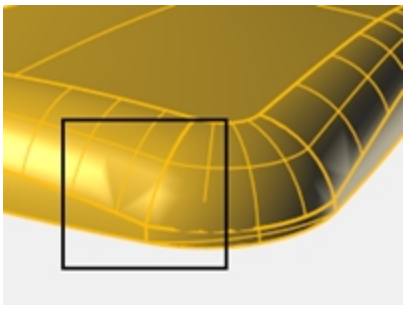
1. Use the **CreateSolid** command *(Solid menu: Create Solid)* to join and trim the surfaces into a closed solid.
2. Use the **FilletEdge** command *(Solid menu: Fillet Edge > Fillet Edge)* to fillet the edges.
3. Set the **CurrentRadius** to **15**, select the four vertical edges, and press Enter to make the fillets.



4. Repeat the **FilletEdge** command to fillet the top edges.
5. Set the **CurrentRadius** to **10**, select the eight top edges, and press Enter to make the fillets.
   The resulting surface is very clean and smooth with no hard edges.



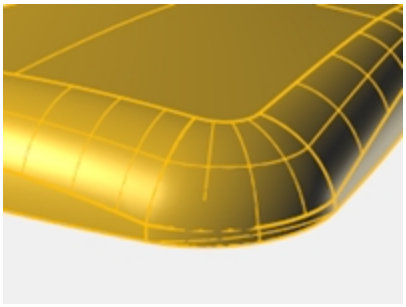
Note: You may notice a defect in a shaded viewport at one or more of the corners. This is a render mesh related defect. There is nothing wrong with the geometry.



Fix the mesh

1. Use the **Options** command to change the mesh settings.
2. On the **Mesh** page, change to a **Custom** mesh.
   Use the settings below.

The visual defect goes away.



Chapter 8 - Advanced surfacing concepts

There are an infinite number of complex and tricky surfacing problems. In this chapter, we will look at several tricks that help in getting certain types of surfaces built cleanly. The goal, apart from showing you a few specific techniques used in these examples, is to suggest ways in which the Rhino tools can be combined creatively to help solve surfacing problems.

In this chapter, you will learn to make soft domed button shapes, creased surfaces, and how to use curve-fairing techniques.

Dome-shaped buttons

The surfacing goal in this exercise is to create a dome on a shape like a cell phone button where the top must conform to the general contour of the surrounding surface but maintain its own shape as well. There are a number of ways to approach this; we will look at three methods.

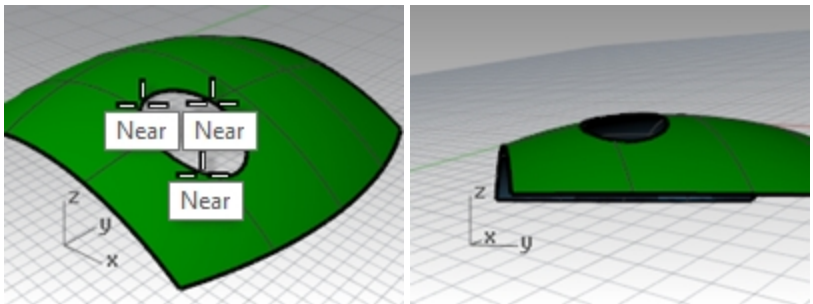
Exercise 8-1 Soft domed buttons

Open and prepare the model

1. Open the model **Button Domes.3dm**.
   The key to this exercise is defining a custom construction plane that represents the closest plane through the area of the surface that you want to match. Once you get the construction plane established, there is a variety of approaches available for building the surface.
   There are several ways to define a construction plane. In this exercise, we will discuss four methods: construction plane through three points, construction plane perpendicular to a curve, construction plane tangent to a surface, and fitting a plane to an object.
2. Use **OneLayerOn** to turn on the **Surfaces** to **Match layer** to see the surface that determines the cut of the button.

Create a custom construction plane using three points method

1. Turn off modeling aids ortho and grid snap.
2. Start the **CPlane** command with the **3Point** option *(View menu: Set CPlane > 3 Points)*.
3. In the **Perspective** viewport, using the **Near** object snap, pick three points on the edge of the trimmed hole.
   The construction plane now goes through the three points. Notice the construction plane origin is at the first point.

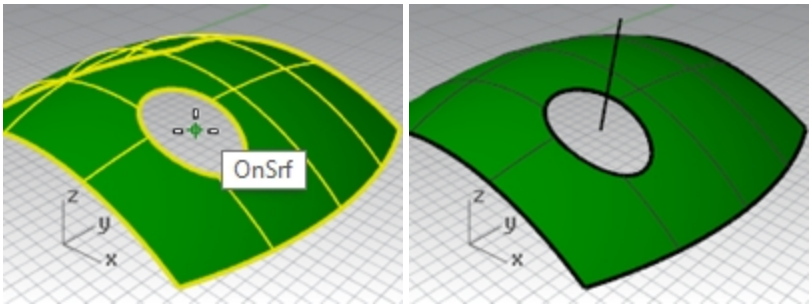


4. Rotate the **Perspective** viewport to see the grid aligned with the trimmed hole.

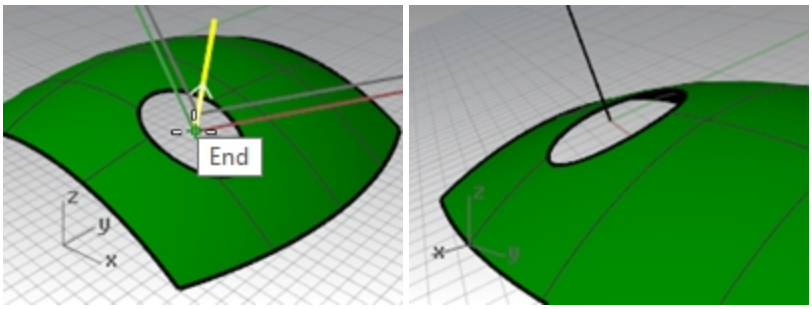
Create a custom construction plane perpendicular to a curve

With a line normal to a surface and a construction plane perpendicular to that normal line, you can define a tangent construction plane at any given point on the surface

1. Start the **CPlane** command with the **Previous** option *(Viewport title right-click menu: Set CPlane > Undo CPlane Change)*.
2. Use the **Line** command with the **Normal** option *(Curve menu: Line > Normal to Surface)* to draw a line normal to the underlying surface at a point near the center of the trimmed hole.



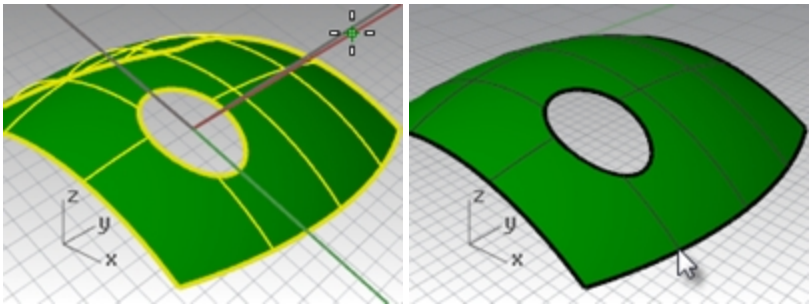
   Set the command-line option for **IgnoreTrims=Yes** so that the line can be drawn from a point inside the trimmed hole in the surface.
3. Start the **CPlane** command with the **Curve** option *(View menu: Set CPlane > Perpendicular to Curve)*.
4. Pick the normal line.
5. Use the **End** object snap and pick the end of the normal line where it intersects the surface.
   The construction plane is set perpendicular to the normal line.

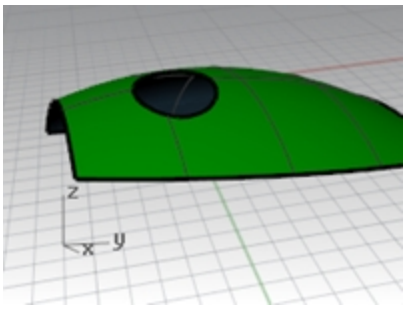
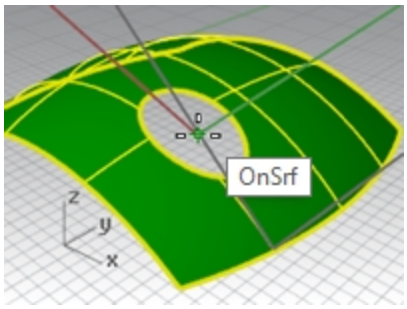


Create a custom construction plane to a surface

This function sets the construction plane to match a surface. The placement is constrained so that the construction plane is tangent to the surface at any given point on the surface. This works like the previous method without the need to make the normal line.

1. Set the **CPlane** to the previous option *(Viewport title right-click menu: Set CPlane > Undo CPlane Change)*.
2. **Delete** the normal line.
3. Start the **CPlane** command with the **Surface** option *(View menu: Set CPlane > Origin)*.
4. Select the surface.
5. For the **CPlane** origin, change the **IgnoreTrims** option to **Yes**, then pick a point near the center of the hole.
6. For the **X axis direction**, pick a point in the direction of the long dimension of the trim curve.
   The construction plane is set tangent to the surface at the origin.

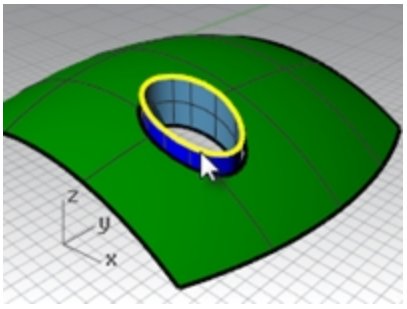


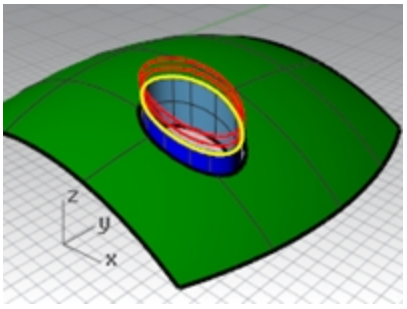
Option 1 - Use a loft surface to make the button

Using the **PlaneThroughPt** command to create a surface through a sample of extracted point objects will generate a plane that best fits the points. The **CPlane** command with the **Object** option places a construction plane with its origin on the center of the plane. This is a good choice in the case of the button in this file. There are several curves from which the points can be extracted—the edge of the button itself, or from the trimmed hole in the surrounding surface.

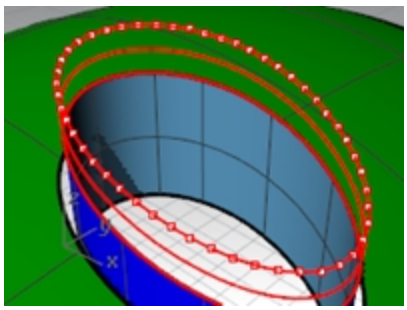
1. Start the **CPlane** command with the **Previous** option *(Viewport title right-click menu: Set CPlane > Undo CPlane Change)*.
2. Turn on the **Surfaces** and the **Curves** layer. Make the **Curves** layer the current layer.
3. Start the **DupEdge** command *(Curve menu > Curve from objects > Duplicate edge)* to duplicate the top edge of the button surface, press Enter.



4. **Copy** the duplicated curve vertically twice, about .5mm apart.
   The vertical position of these curves will determine the shape of the button.

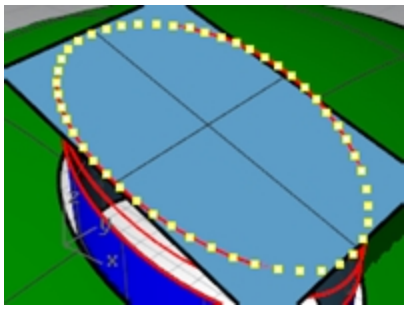
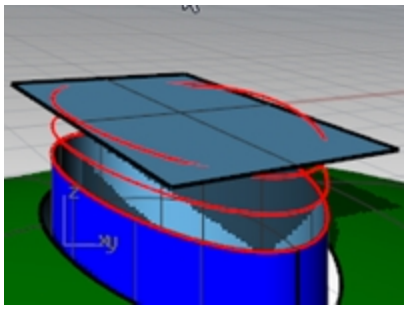


5. Use the **Divide** command *(Curve menu: Point Object > Divide curve by > Number of segments)* to mark off the top copied curve with **50** points.
   Set the command-line options to **Split=No** and **GroupOutput=Yes**.

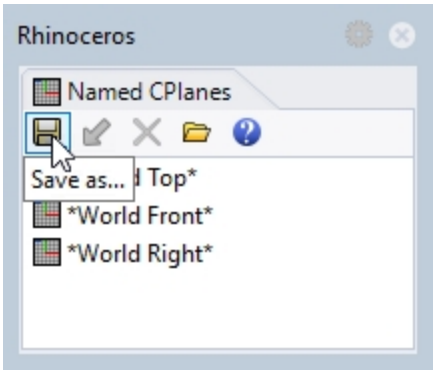


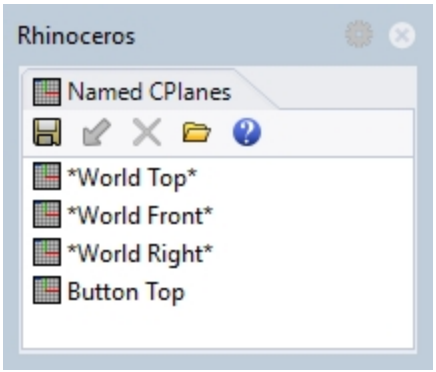
6. Use **SelLast** to select the points just created.
7. Use the **PlaneThroughPt** command *(Surface menu: Plane > Through Points)* with the selected points.

8. Press the Delete key to delete the point objects that are still selected.
   A rectangular plane is fit through the selected points.

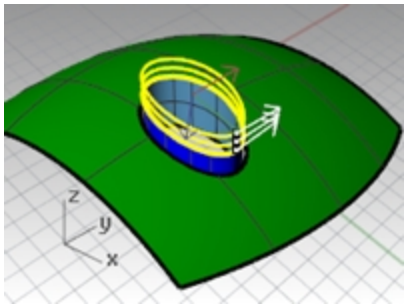
9. Use the **CPlane** command with the **Object** option *(View menu: Set CPlane > To Object)* to align the construction plane with the plane.
10. On the **View** menu, select **Set CPlane**, click **Named CPlanes**, then click the **SaveAs** icon to save and name the custom construction plane **Button Top**.
    This will allow you to restore this custom construction plane at any time.
11. **Delete** the surface you used to create the **Button Top** construction plane.



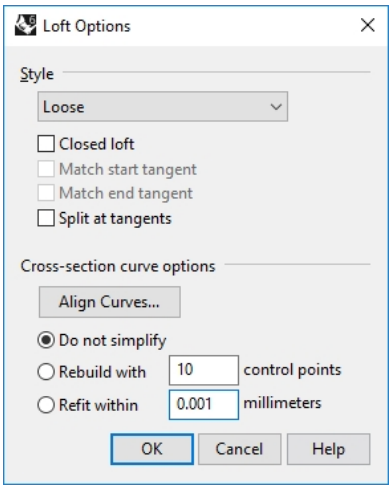


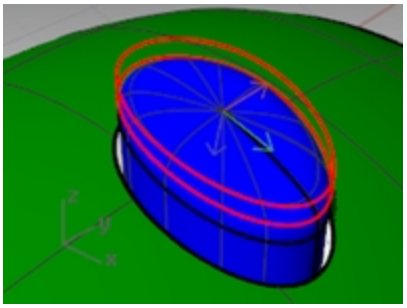
Loft the button

1. Set the **Surfaces** layer as the current layer.
2. Use **Loft** to make the button.
3. Select the top edge of the surface and the two copied curves.
4. After selecting the curves, on the command-line, click the **Point** option.
5. For the **Loft** endpoint, make sure the view that has the custom construction plane is the current view, then type **0** (zero) and press Enter.
   The loft will end at a point in the middle of the plane, which is the origin of the construction plane.

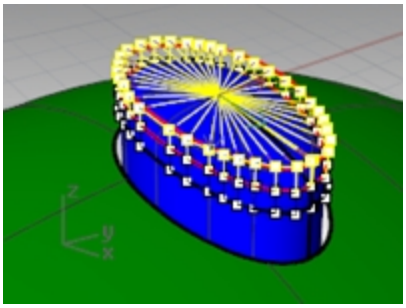


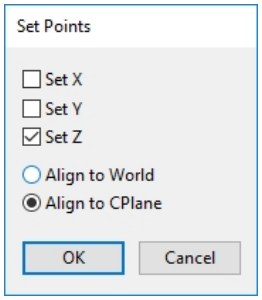
6. In the **Loft Options** dialog box, under **Style**, choose **Loose**, click OK.
   With the **Loose** option, the control points of the input curves become the control points of the resulting surface, as opposed to the **Normal** option, in which the lofted surface is interpolated through the curves.



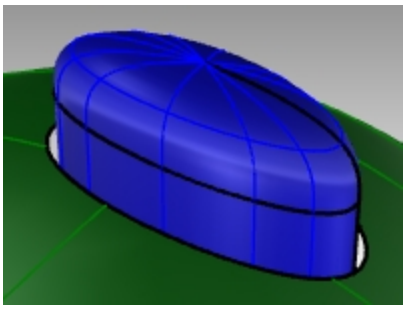


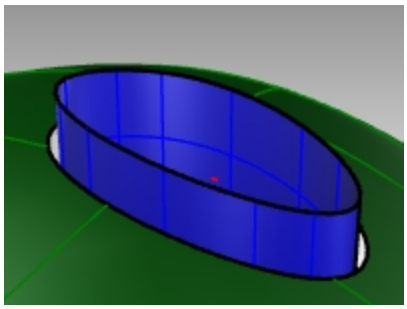
7. Turn on control points on the lofted surface.
8. To select the next ring of points out from the center, select one point, and then use **SelV** or **SelU** to select the whole ring of points.
9. Use the **SetPt** command *(Transform menu: Set XYZ Coordinates)* to set the selected control points to the same CPlane Z elevation as the singularity in the center of the surface by snapping to the point at the singularity.
   You have a custom CPlane active and you can set all the Z values of the points, relative to the CPlane. The Z values will be the same as the middle point.
   Un-check the Set X and Set Y options (you can right click on 'Set Z' to quickly clear the others and set that one).
10. In the dialog box, set the **Align to CPlane** option.
    Remember, this elevation is relative to the current construction plane.





11. With the **Point** osnap active, snap to the point in the middle of the loft, which is the point at the singularity.





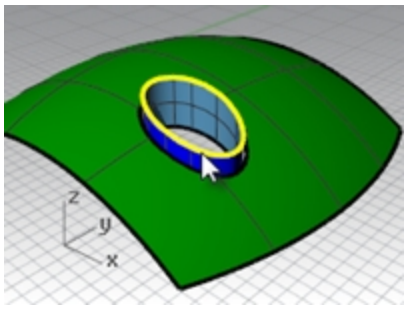
Note: You can use **History** when creating the loft, in which case the **SetPt** operation should be applied to the topmost curve in the loft, not to the loft surface control points.

12. Hide the button top surface.

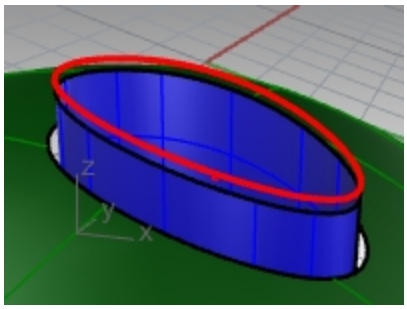
Option 2 - Use a patch surface to make the button

Next, you will create the button surface with the **Patch** command. Patch also support History. If Record History is on when the **Patch** command is used to create the button surface, changing the input surface will update the Patch surface.

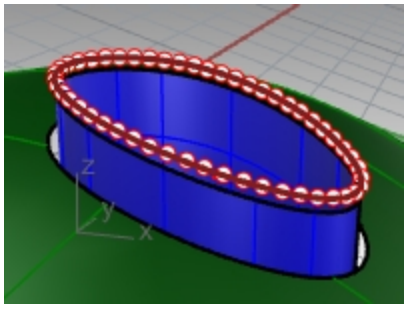
1. Use the **DupEdge** command to duplicate the top edge of the surface.



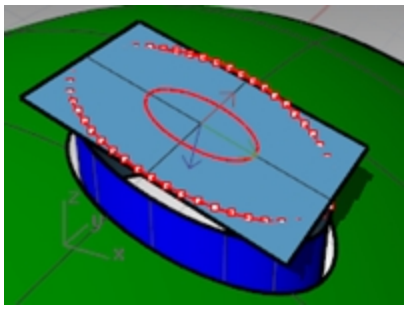
2. Move the duplicated curve up in the World z-direction by a small amount.



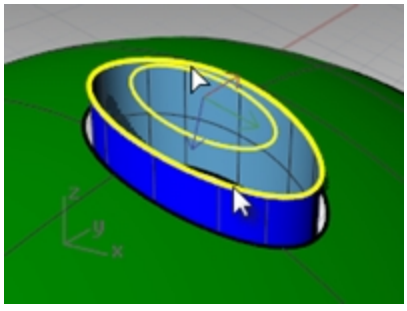
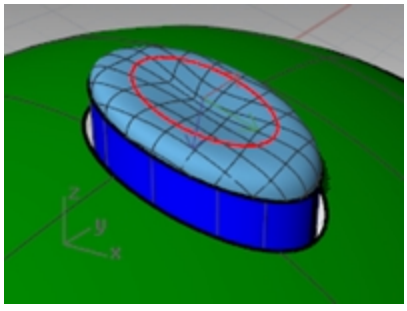
3. Use **Divide** to mark off this curve with **50** points as before.



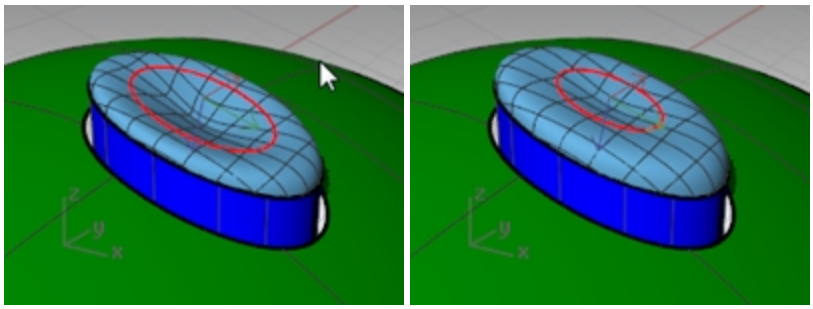
4. Use the **PlaneThroughPt** command with the selected points, and then delete the points like the previous exercise.
5. Use the **CPlane** command with the **Object** option to set the construction plane to the planar surface.
6. Make a circle or ellipse centered on the origin of the custom construction plane.



7. Use the **Patch** command, selecting the top edge of the button and the ellipse or circle.
   The surface is tangent to the edge and concave on the top.

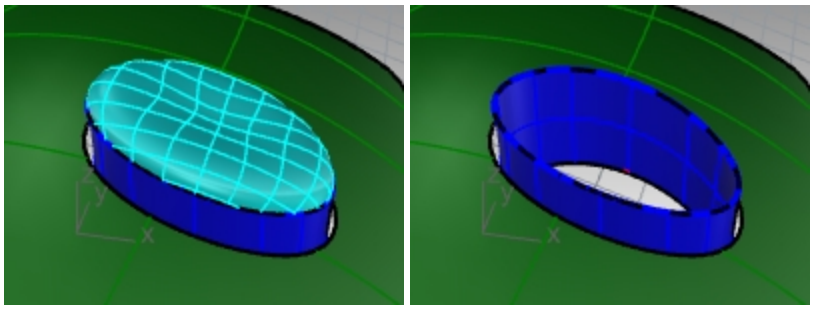



   The size and vertical position of the circle/ellipse will affect the shape of the surface.



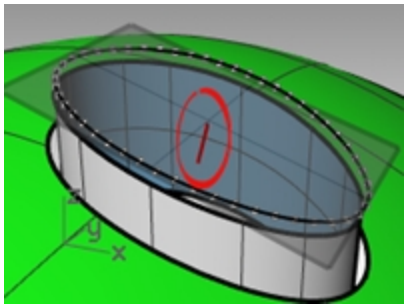
Note: If you recorded **History** for the patch, you can select the ellipse and move it up and down or scale it in two dimensions to modify the patch shape.
The **Gumball** control is perfect for making these adjustments. Make sure the **Gumball alignment** is set to **CPlane**.

8. Hide this button top surface.

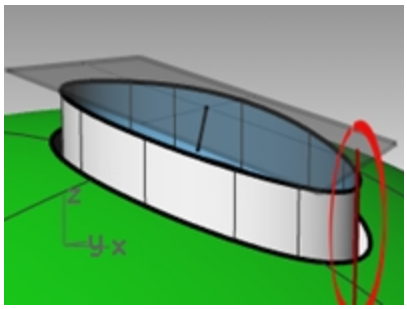


Option 3 - Use a rail revolve with line as the revolve axis and match surface

1. Use the **DupEdge** command to duplicate the top edge of the surface.
2. **Move** the duplicated curve in the World Z-direction a small amount.
3. **Divide** this curve with 50 points.
4. Create a surface with**PlaneThroughPt** as before. Delete the points like the previous exercise.
5. Use the **CPlane** command with the **Object** option to set the construction plane to the planar surface.
6. Use the **Lock** command (*Edit menu: Visibility >Lock*) to lock the surface created with **PlaneThroughPt**.
7. Use **Line** with the **Vertical** option to make a line of any convenient length from the origin of the construction plane down towards the button surface.

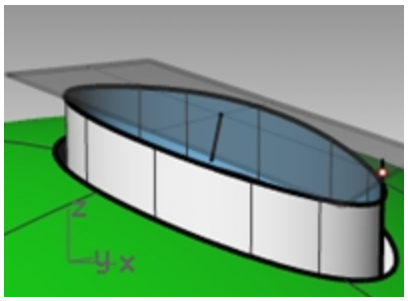


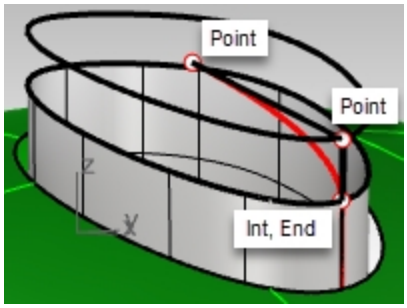
8. Use the **Extend** command *(Curve menu: Extend Curve > By Line)* to extend the edge at the seam through the rectangular surface. (If no seam is available use ExtractIsocurve to create a curve from the surface and extend.)



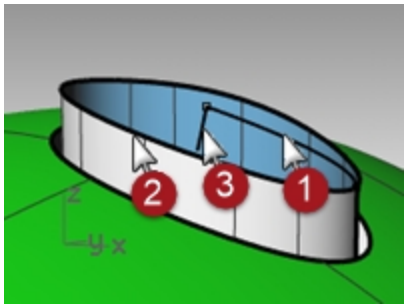
9. Use the **Intersect** command *(Curve menu: Curve From Objects > Intersection)* to find the intersection between the extended line and the rectangular surface.

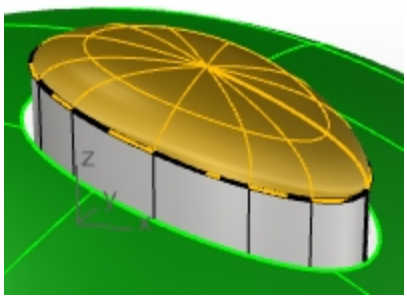
10. Use the **Curve** command to draw a curve from the end of the normal line, using the intersection point as the middle control point, to the end of the seam to use as a profile curve.





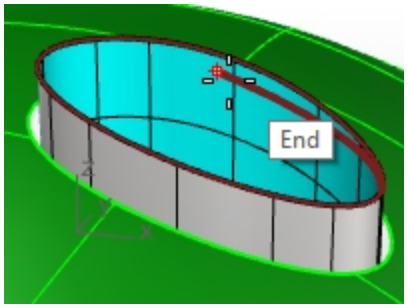
11. Start the **RailRevolve** command *(Surface menu: Rail Revolve)*. Set the **ScaleHeight** option to **Yes**.
In general, setting ScaleHeight to Yes is helpful in this case where the rail curve is not planar. You will select the top edge of the extruded button as the rail.
12. **Select** the curve you just created (1) as the profile curve, the top edge of the surface (2) as the path curve. Select the upper end of the vertical line (3) as one end of the revolve axis and the lower end as the other end of the revolve axis.



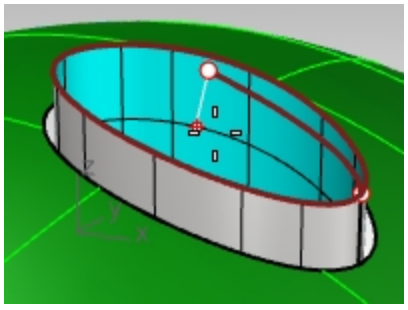


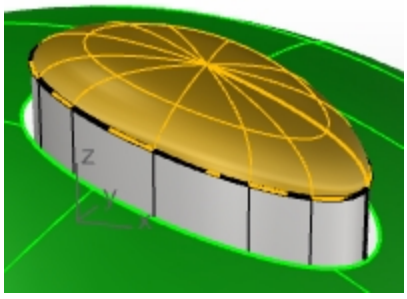
Next, the second point is directly vertical from the origin, and relative to the CPlane, (not the World cplane).

Optional Elevator Mode: Instead of creating the vertical line and picking the end points to specify the start and end of the revolve axis, use **Elevator Mode.**
At the prompt for **Start of Revolve Axis**, pick the end point of the profile curve with the **End** osnap.

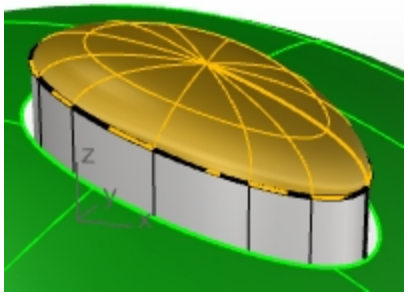
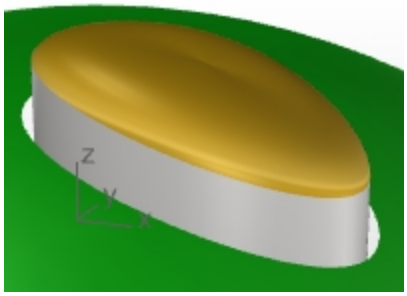


Then hold down the Control key, again pick on the same end point on the profile curve.
Next, drag the cursor any distance away and pick the second point with the help of Elevator mode. This will input the revolve axis without your having to create a line.

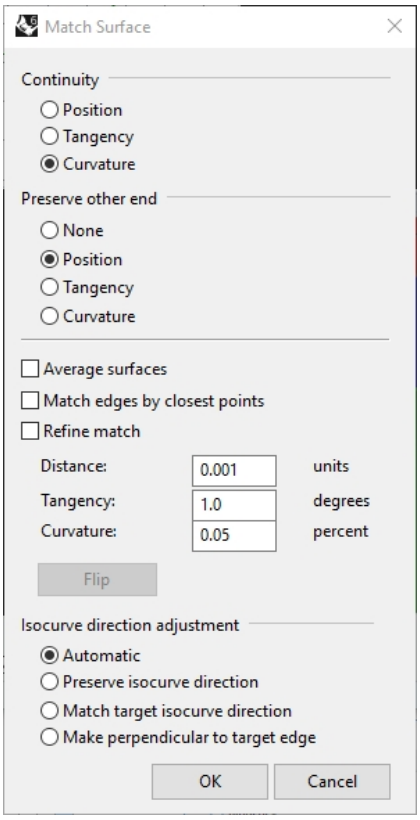


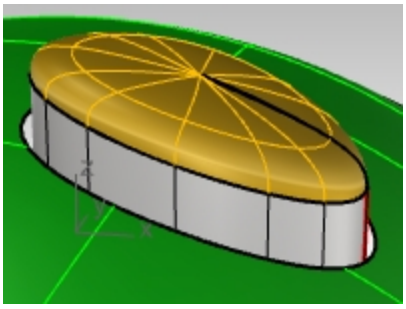


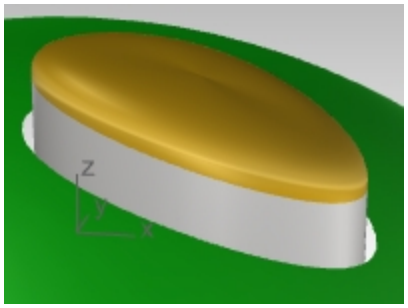
13. **RailRevolve** does not pay attention to continuity during the surface creation.

You will need to match the new surface to the vertical sides of the button for tangency or curvature with the **MatchSrf** command .
In the **MactchSrf** dialog, set **Continuity** to **Curvature,** set **Preserve other end** to **Postion**, and set **Isocurve direction adjustment** to **Automatic.**

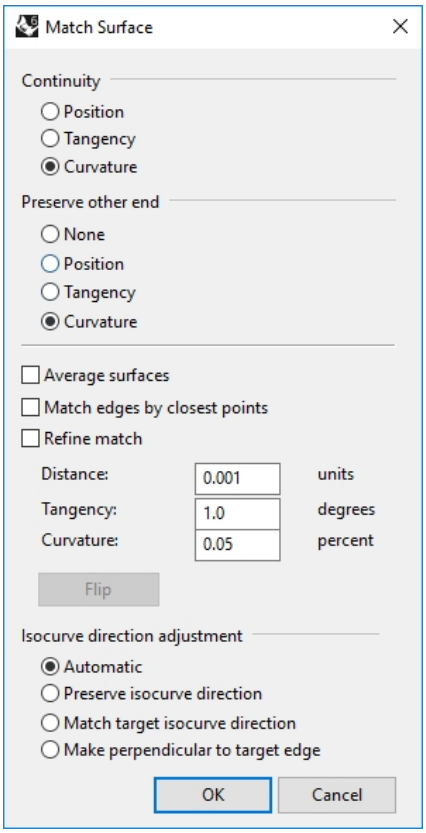


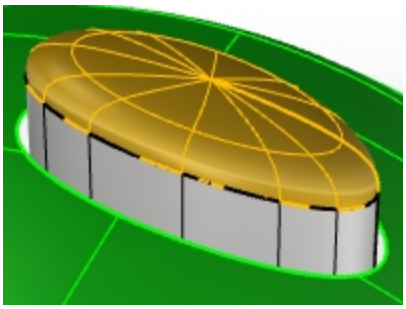


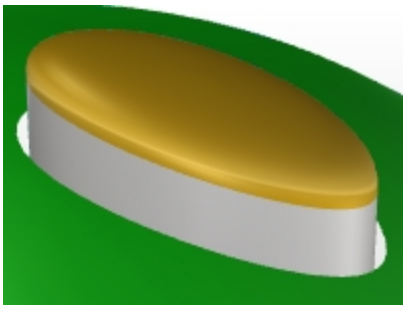


On close inspection, this seems to pinch at the top. **Undo** and try this **Match** again.

14. In the **MactchSrf** dialog, set **Continuity** to **Curvature**, set **Preserve other end** to **Curvature,** and set **Isocurve direction adjustment** to **Automatic.**







The continuity of the button surface looks better after MatchSrf with these settings.

Creased surfaces

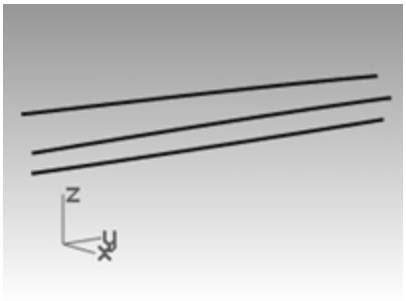
Often a surface needs to be built with a crease that may start at a particular angle and change to another angle or diminish to zero. The crease in the car body in the image is an example of this. The following exercise covers two possible situations.

Exercise 8-2 Surfaces with a crease (part 1)

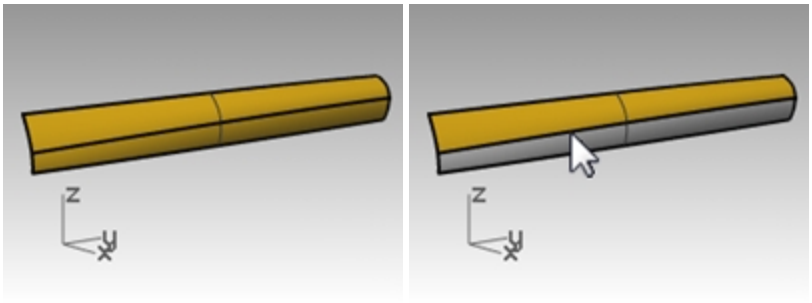
The key to the following exercise is to get two surfaces that match with different continuity at each end. At one end, we will match the surface with a 10-degree angle, and at the other end, we will match the surface with tangency continuity. To accomplish this we will create a dummy surface at the correct angles and use this to match the lower edge of the upper surface. When the dummy surface is deleted or hidden, the crease appears between the two surfaces we want to keep.

Open and prepare the model

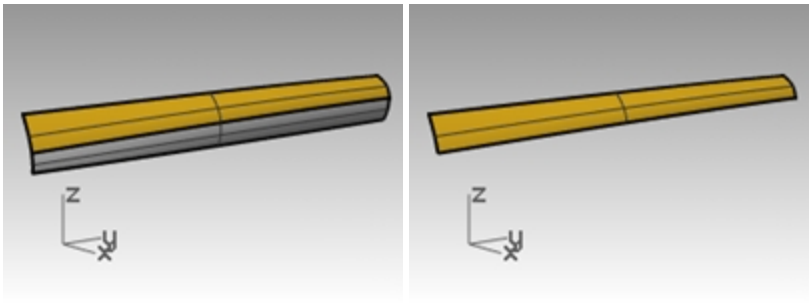
1. **Open** the model **Crease 01.3dm**.
2. Turn on the **Curve** and **Loft** layers.
3. Make the **Loft** layer current.
4. Use the **Loft** command to make a surface from the three curves.
   The **Loft** command remembers the settings across sessions. Make sure the **Loft style** is set to **Normal** and **Do not simplify**.



5. We are going to make a surface that includes all the curves but has a crease along the middle curve.
   Use the middle curve to **Split** the resulting surface into two pieces.
6. Use the **ShrinkTrimmedSrf** command *(Surface menu: Surface Edit Tools > Shrink Trimmed Surface)* on both surfaces.



   With a surface that results from a split at an isocurve, shrinking it will allow the edge to be an untrimmed edge because the trim corresponds to the natural untrimmed surface edge.
   By trimming with a curve used in the loft, the curve is in effect an isocurve.
   You can also use the Isocurve option in the Split command when the object to be split is a single surface.



7. **Hide** the lower surface and turn off the **Curve** layer.

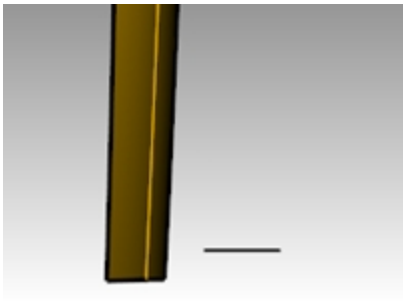
Create the dummy surface

We will change the top surface by matching it to a new dummy surface.

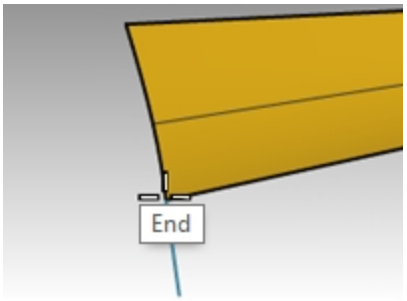
The dummy surface will be made from one or more line segments along the bottom edge of the top surface that are set at varying angles to it.

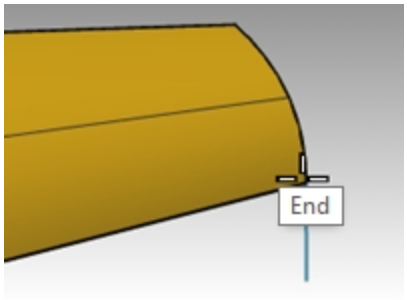
To get a line that is not tangent but is at a given angle from tangent, the easiest method is to use the transform tools to place the line tangent and then to rotate it by the desired increment.

1. Make the **Dummy Curve** layer current.
2. In the **Top** viewport, draw a line **20** units long.

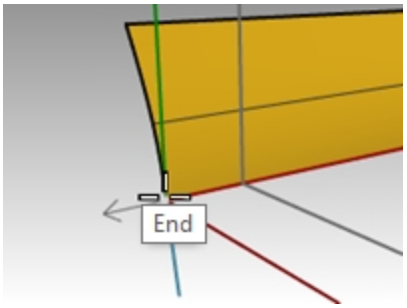


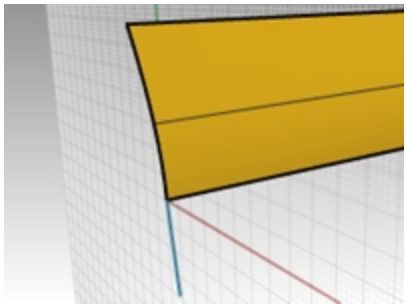
3. Start the **OrientCrvToEdge** command *(Transform menu: Orient > Curve To Edge)*.
4. For the **Curve to orient**, select the line.
5. For the **Target surface edge**, select the lower edge of the surface.





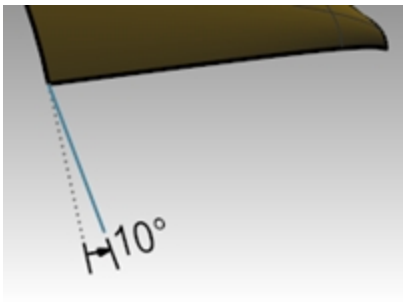
6. For the **Pick target edge point**, change the command-line option to **Copy=Yes**, and snap to an endpoint of the edge.
7. For the **Pick target edge point**, snap to the other endpoint.
8. Press Enter.
   The result should look like the images above.
9. Next you will set a construction plane perpendicular to the lower edge of the surface.
   In the **Perspective** viewport, right-click the **Viewport title**and off the menu select **Set CPlane** andthe **Perpendicular to curve** option. Snap to the left end point of the lower edge for the construction plane origin.



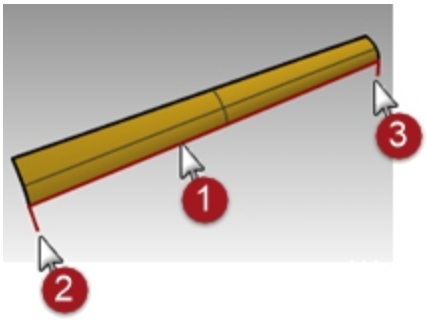


Set up the rail line

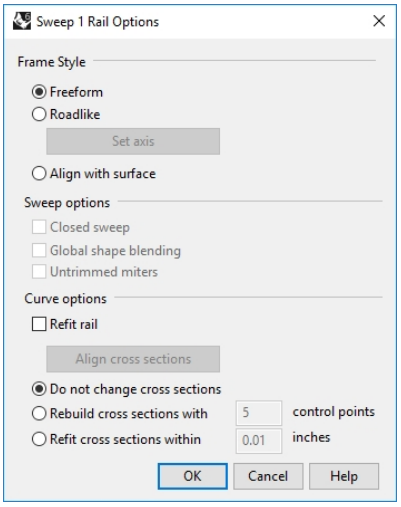
1. **Select** the line segment at the left end and start the **Rotate** command.
2. Set the center of rotation at the origin of the new custom construction plane.
3. Rotate the segment 10 degrees.
   The result should be like the image on the right.

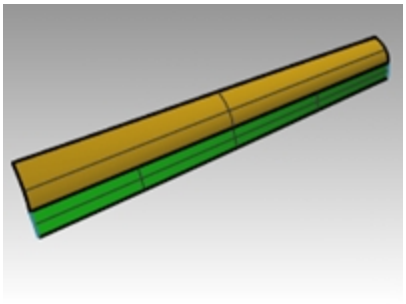


4. Make the **Dummy Surface** layer current.
5. Use the **Sweep1** command *(Surface menu: Sweep 1 Rail)* to create the dummy surface.
6. **Select** the lower edge of the upper surface (1) as the rail and the two line segments (2 & 3) as cross-section curves.
   Make sure to use the surface edge and not the original input curve as the rail for the sweep.



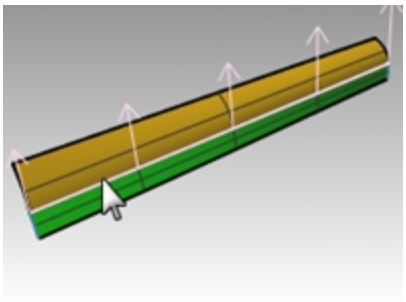
7. In the **Sweep 1 Rail Options** dialog box, under **Style**, choose **Align with surface**.
   This option causes the sweep surface to derive from the cross section curves its angle relative to the base surface at its edge. A shape curve tangent to the base surface will hold that tangency as it sweeps along the edge, unless another shape curve with a different angle to the surface is encountered, in which case there will be a smooth transition from one to the next



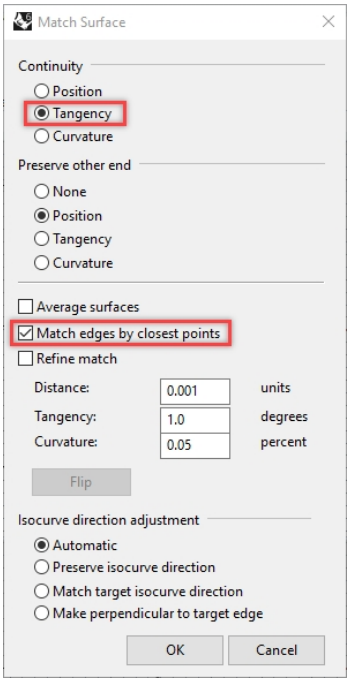


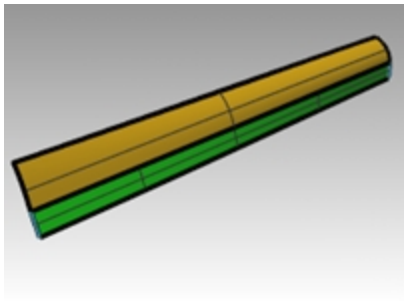
Match the surface to the dummy surface

1. Use the **MatchSrf** command to match the upper surface to the dummy surface.
2. **Select** the lower edge of the upper surface.

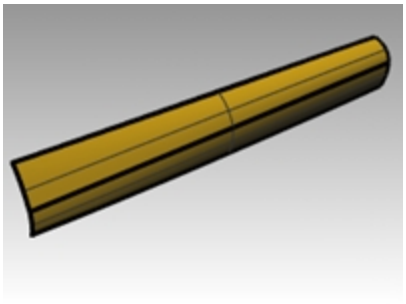


3. **Select** the upper edge of the dummy surface.
4. In the **Match Surface** dialog box, choose **Tangency** and check the **Match edges by closest points** option. This will keep distortion to a minimum.

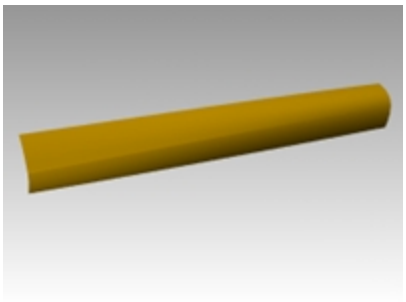




5. **Show** the lower surface and hide the dummy surface.



6. **Join** the lower surface with the upper surface.
   Because the surfaces are untrimmed, you have the option to merge the surfaces back into one surface. The crease fades smoothly from one end to the other of the polysurface. If more control is needed over the angles of the crease, more segments can be placed to create the dummy surface.



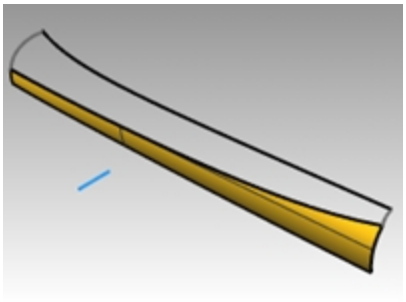
Surfaces with a crease - Part 2

In this exercise, there is no convenient relationship between the crease curve and the surface. While similar to the other example, the upper surface is made with a two-rail sweep.

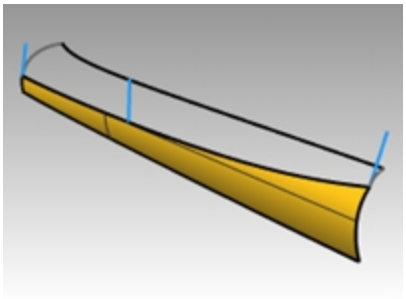
Exercise 8-3 Surfaces with a crease (Part 2)

Create a crease with trimmed surfaces

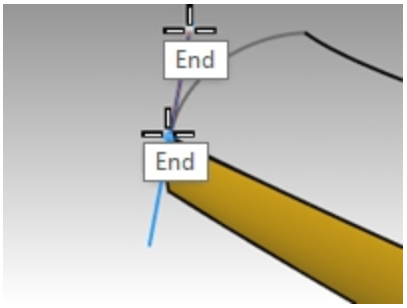
1. **Open** the model **Crease 02.3dm**.
2. Use the **Line** command *(Curve menu: Line > Single Line)* to draw a single line anywhere in the **Top** or **Perspective** viewport.
   We will use this line to make a dummy surface.

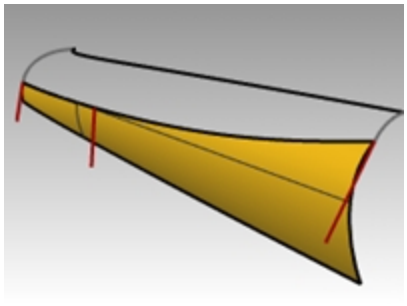


3. Use the **OrientCrvToEdge** command *(Transform menu: Orient > Curve to Edge)* to copy the curve for the dummy surface to the upper edge of the lower surface.
4. Place a line at each end of the edge and somewhere in the middle of the edge.

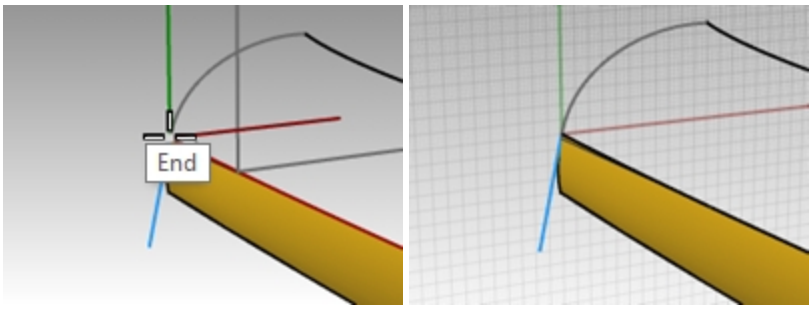


5. **Move** each line segment by moving its upper end to the lower end of the same segment.

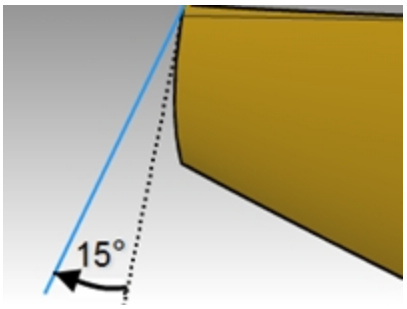




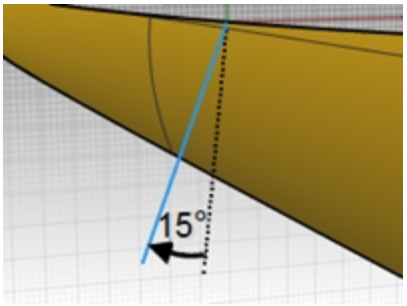
6. Use the **CPlane** command *(View menu: Set CPlane > Perpendicular to Curve)* to set the construction plane to align with the line at the left of the surface.



7. Use the **Rotate** command *(Transform menu: Rotate)* to rotate the line **15** degrees as shown in the illustration on the right.

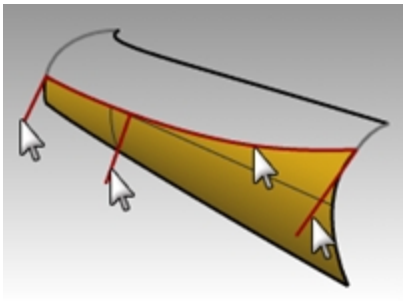


8. Repeat these steps for the line in the middle of the surface.

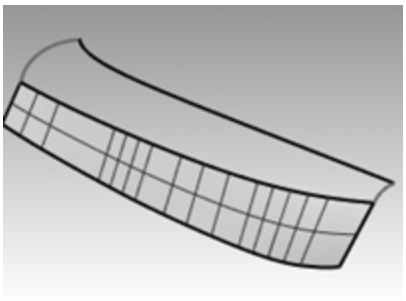


Make the dummy surface

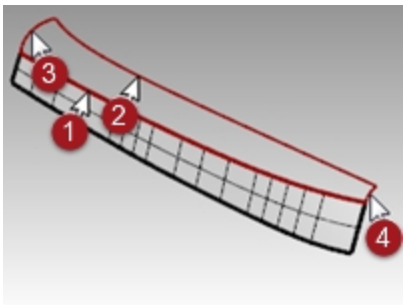
1. Use the **Sweep1** command to create the dummy surface.
2. **Select** the upper edge of the lower surface as the rail and the three line segments as cross-section curves. Use the **Align with surface** style for the sweep.



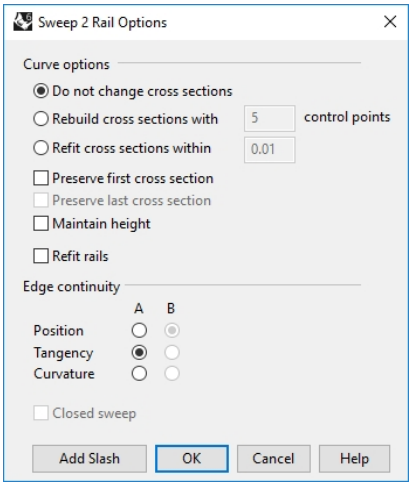
3. **Hide** the original surface.



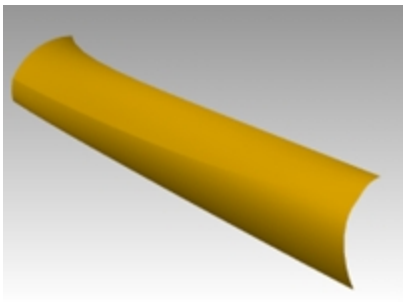
4. Use the **Sweep2** command to make the upper surface.
   Select the upper edge of the dummy surface as a rail (1) and the long curve at the top as the other rail (2).
   Select the curves at both ends as the cross-section curves (3) and (4).



5. In the **Sweep 2 Rails Options** dialog box, for the **Rail continuity of edge A**, choose **Tangency**.



6. **Delete** the dummy surface.
7. Use **Show** or **Show Selected** *(Edit menu>Visibility>Show selected)* to show the original lower surface.
8. **Join** the lower surface with the upper surface.



Input curve fairing to control surface quality

Curves in Rhino can come from many sources, they can be:

- Created in Rhino directly
- Imported from digitized data
- Imported from another application
- Section curves generated from a mesh

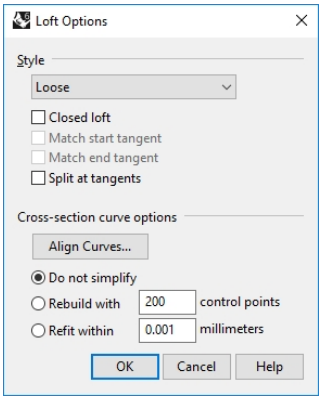
It is important to understand that many of these curves need to be optimized for quality.

Fairing is a technique used to simplify curves while improving their curvature graphs and keeping their shape within tolerance. It is especially important to fair curves that are generated from digitized data, intersections, extracted isocurves, or curves from two views.

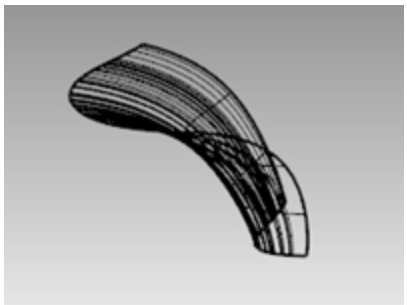
Generally, curves that are single-span curves work better for this process. A single span curve is a curve that has one more control point than the degree. For examples a degree 3 curve with 4 control points, a degree 5 curve with 6 control points, or a degree 7 curve with 8 control points.

Analyze a lofted surface with curvature analysis

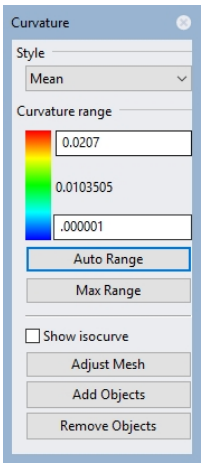
1. **Open** the model **Fair Curves.3dm**.
2. **Select** the curves and use the **Loft** command *(Surface menu: Loft)* with **Style** set to **Normal** and **Cross-section curve** options set to **Do not simplify** to make a surface.



The surface is very complex. It has many more isocurves than are needed to define the shape, because the knot structures of the curves are very different.
The surface also has compound curvature.

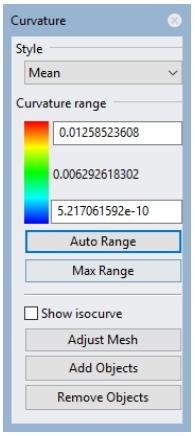


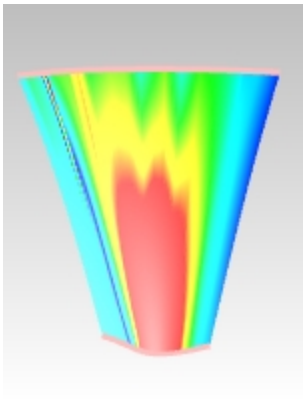
3. **Select** the lofted surface and start the **CurvatureAnalysis** command *(Analyze menu > Surface>Curvature analysis)*.

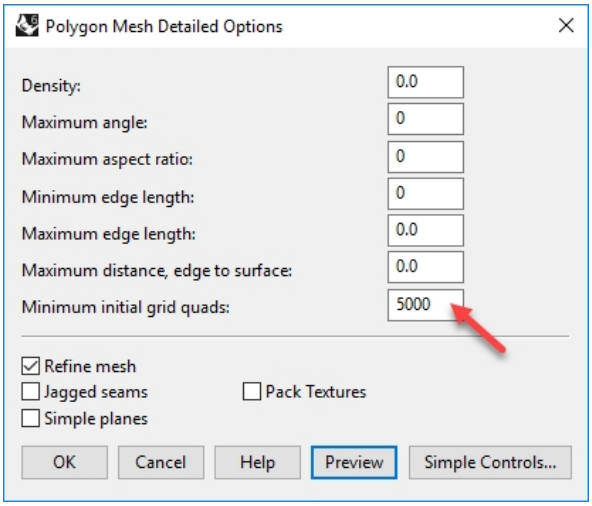


This creates a false color display using the same type of analysis meshes as the **Zebra** command.
The amount of curvature is mapped to a range of colors allowing you to analyze for areas of abruptly changing curvature or flat spots.
Choose **Mean** from the **Style** drop down.
This style is useful for showing discontinuities in the curvature—flat spots and dents. The mean is between the two curvature circle values at each point, mapped to a color value.

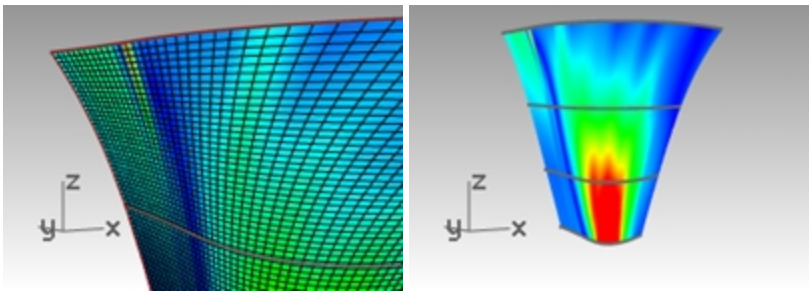
4. Click the **AutoRange** button.
5. Click the **Adjust Mesh** button and adjust the **Minimum initial grid quads** to have at least **5000** minimum grid quads to ensure a smooth display of the color range.







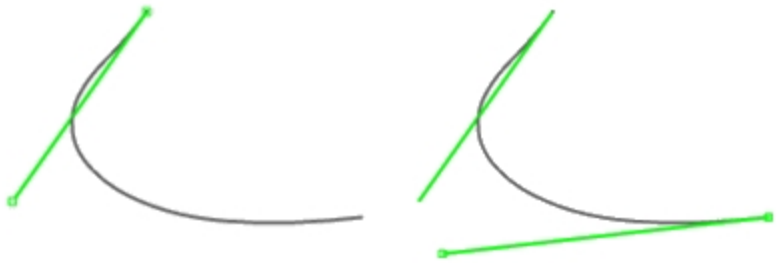
Note the streaky and inconsistent colors on the surface. This indicates abrupt changes in the surface.



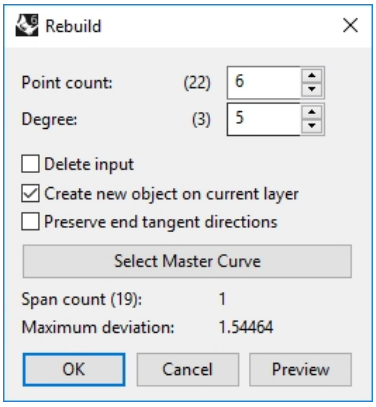
6. **Undo** or delete the lofted surface.

Rebuild the curves

1. Change to the **Tangency Direction** layer.
2. Use the **Line** command *(Curve menu: Line > Single Line)* with the **Extension** option, to make a line that maintains the tangency direction of an original curve from each end point and coming back towards the curve, any length. Make the lines long enough to cross one another.

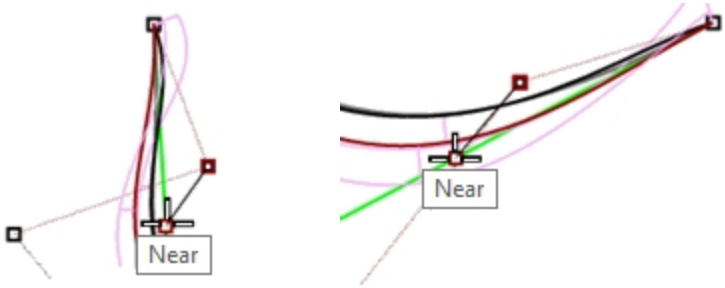


3. Change to **Rebuilt Curves** layer, and **Lock** the **Tangency Direction** layer.
4. Use the **Rebuild** command *(Edit menu: Rebuild)* to rebuild the curve.
5. Although there is a **Rebuild** option in the **Loft** command, rebuilding the curves before lofting them gives you control over the degree of the curves as well as the number of control points.
6. In the **Rebuild Curve** dialog box, change the **Degree** to **5** and the **Point Count** to **6** points.
7. Clear the **Delete input** option and check **Create new object on current layer** option.
8. Click the Preview button. Note how much the curves deviate from the originals.

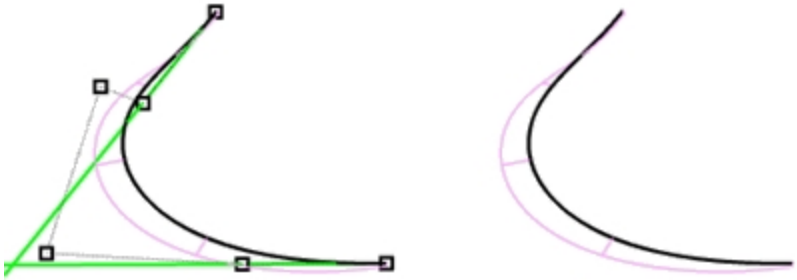


This makes the curves into single-span curves. Single-span curves are Bézier curves. A single-span curve is a curve that has degree +1 control points. While this is not necessary to get high quality surfaces, it produces predictable results.

9. Lock the **Original Curves** layer. We need to see these curves but we do not want to be able to select them.
10. **Select** one of the rebuilt curves, and turn on the control points and **Curvature graph**.
11. Fair the curve by adjusting points until it matches the original curve closely enough.



12. Start by moving the second point from each end of the rebuilt curve onto the tangent line. Use the **Near** object snap to drag along the tangent line.
13. Check the curvature graph to make sure the curve has smooth transitions.
    The curves are fair when the points are adjusted so the rebuilt curves match the original locked curves closely, with good graphs.
14. Fair the other curves the same way.



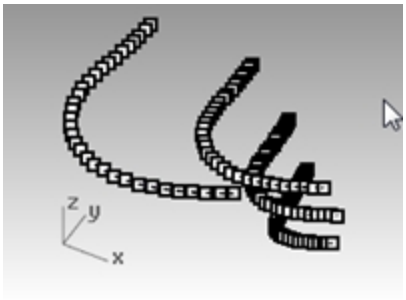
Here are some things to keep in mind when adjusting the curves:

- If you want to keep the curve tangent direction consistent with the original curves, make sure the second point from each curve stays on the green tangent direction line- only move these points with the Near object snap pulling the point onto the line.
- The **DragMode** command, set to **ControlPolygon** will constrain point dragging to the curve control polygon. You can use this tool to keep the tangent directions constant.
- Where possible, when fairing a set of curves to be used as input to a single lofted surface, try to keep the control point arrangement on each curve similar to the neighboring curves. This will help keep the surface nicely aligned.
- When control point adjustment becomes difficult due to the small movements needed, try using the **Nudge** keys to nudge the points by small amounts. See **Help** for more information about **Nudge**.

You can also use the **Gumball** to move points. When fine-tuning with very small point movements needed, you can set **GumballDragStrength** to something less than 100% to allow larger mouse movements to make small changes in the point locations.

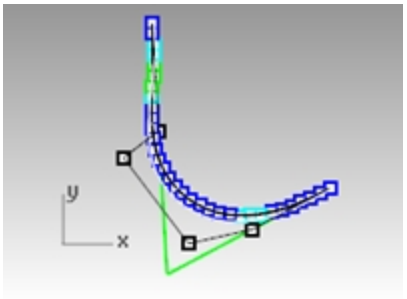
Use PointDeviation to visualize the deviation while you edit the curves

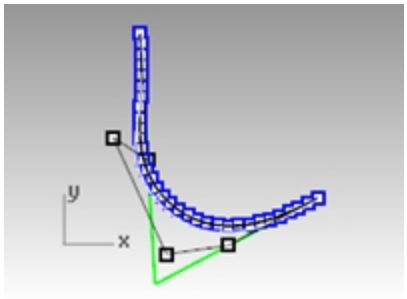
1. Make a **Points** layer and make it current.
2. **Select** all of the original curves and start the **Divide** command *(Curve menu: Point Object>Divide Curve by>Number of Segments)*. Set the **Number of segments** to **32** and **GroupOutput=Yes**.
3. **Deselect** all objects, and select the grouped points.
4. Start the **PointDeviation** command *(Analyze menu: Surface>Point Set Deviation)*, and at the **Select curves, surfaces, and polysurfs to test** prompt, select the, roughly faired, rebuilt curves.



5. When the **Point/Surface Deviation** dialog box pops up, set the numbers as follows:
   Good point = 0.1
   Bad point = 0.5
   Ignore = 1.0
   The display shows the deviation between the points displayed on the original curves and the closest locations to these on the rebuilt curves.
6. **Lock** the **Points** and the **Original Curves** layers.

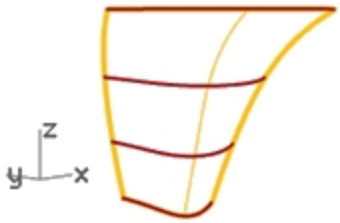
7. Continue to modify the rebuilt curves with the goal being that all points will turn blue, as good points.
   Note: If you close the dialog box, you lose the display and you need to start over with **PointDeviation**.

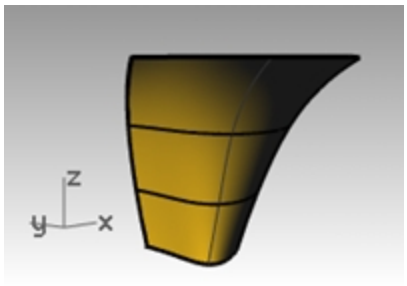




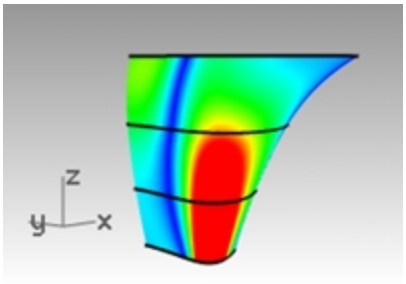
Make a surface with fair curves

1. **Loft** the new curves.
   The shape and quality of the surface has very few isocurves but it is very close to the shape of the first surface.





2. Analyze the surface with **CurvatureAnalysis**.
   Note the smooth transitions in the false color display, indicating smooth curvature transitions in the surface.



Chapter 9 - Modeling from reference images

The **Picture** command is used to bring one or more reference images into a Rhino model.

- A 'picture' is a planar surface with the digital image added to the plane's material as a texture.
- You choose the image that the Picture will display.
- Pictures can be moved, rotated, scaled and otherwise manipulated just like any Rhino object. This makes them very convenient to get them scaled and located exactly where they are needed.

In this exercise we'll bring in two simple sketches with different views of the same object, and we'll place the planes so that their locations and scales make sense with one another and with the scale of the object. Along the way we'll look at some different ways to manage Pictures using layers and materials, and we'll look at some curve and surfacing tools as well.

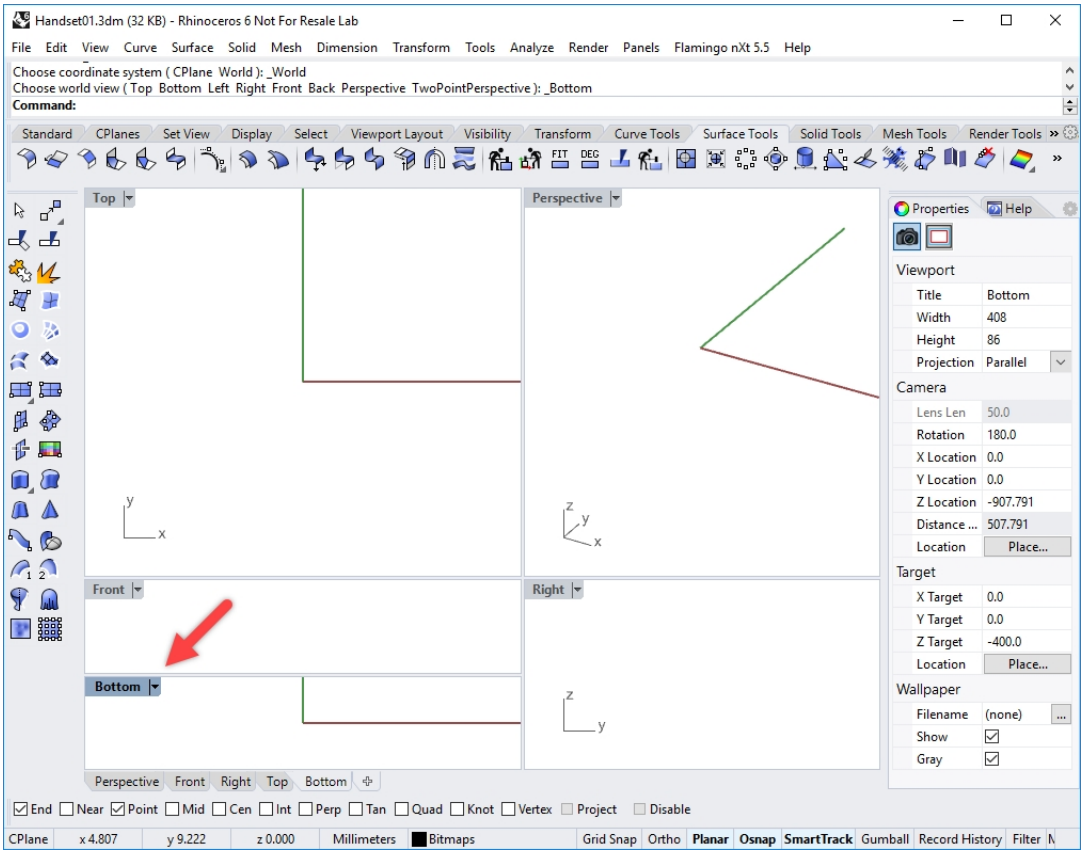
To begin, open the file Handset.3dm. This file does not have any objects in it yet, only some layers set up to save us a bit of time organizing. There is a layer for the Pictures - make that the current layer.

We will begin by taking scanned sketches and placing them in three different viewports. The three hand-drawn images need to be placed in their respective viewports and scaled appropriately so that they match each other.

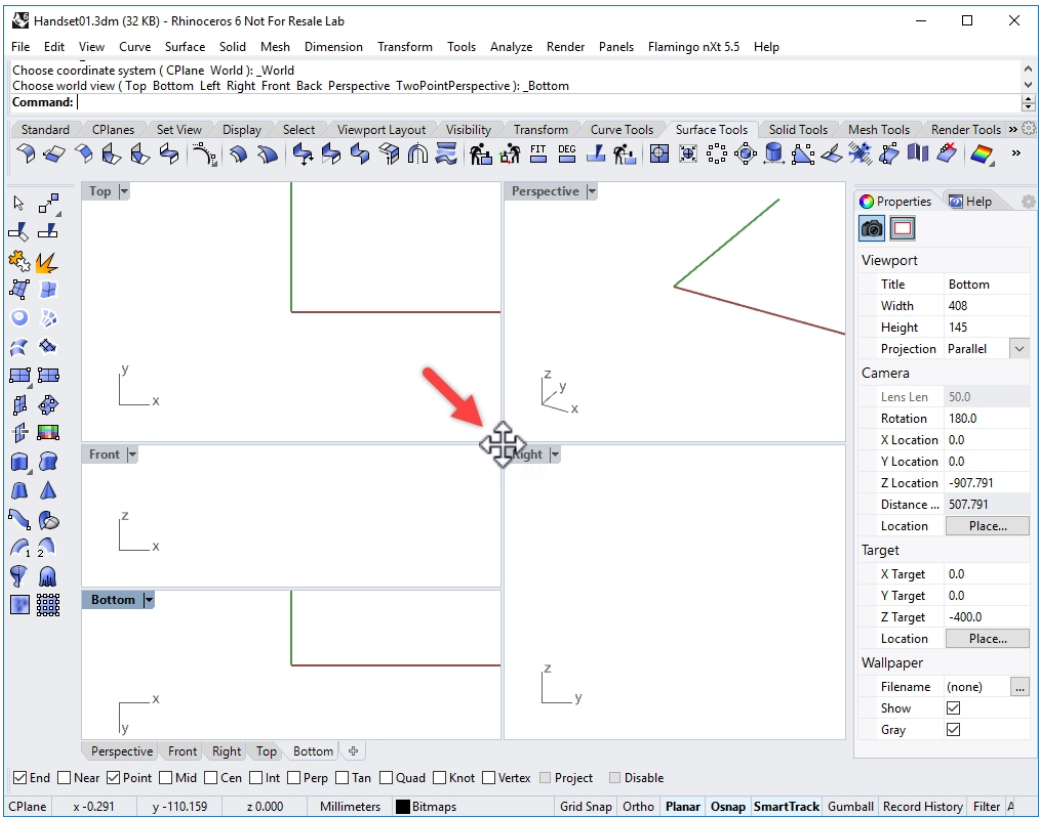
Exercise 9-1 Handset

Open and prepare the model

1. **Open** the model **Handset.3dm**.
2. Drag the center of the Viewports to increase the height of the Top and Perspective viewports to 2/3 the size of the graphics area.



3. Make the **Front** viewport current.
4. From the **View** menu, under **Viewport Layout** click **Split horizontally**.
   This will make a second **Front** viewport.
5. Highlight the lower **Front** viewport. From the **View** menu, under **Set View** click **Bottom** view.
6. Re-align the views if necessary.

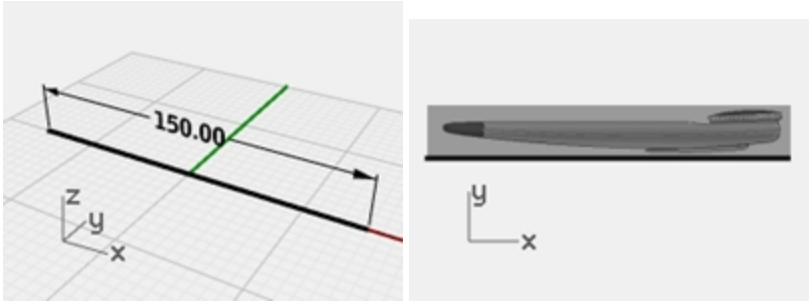


7. In **Rhino Options**, on the **View** page, in the **Viewport properties** section, check the **Linked viewports** option. This allows the viewports to stay aligned with each other when zooming and panning.

Place background bitmaps

We will begin by making reference geometry to help in placing the bitmaps.

1. Draw a horizontal **Line**, from both sides of the origin of the **Top** viewport, **150 mm** long.
2. Toggle the grid off in the viewports that you are using to place the bitmaps by pressing the F7 key.
3. This will make it much easier to see the bitmap.



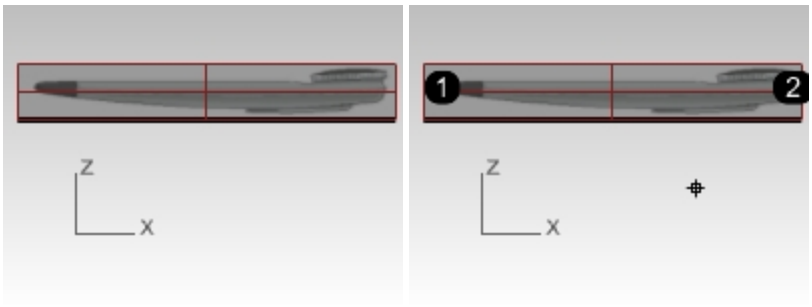
4. In the **Front** viewport, from the **Surface** menu, under **Plane** click **Picture** command.
5. In the **Open Bitmap** dialog, select the **HandsetElevation.bmp**.
6. In the **Front** viewport, using the End osnap, pick the left end of the reference line and then the right end of the reference line.
7. Highlight the picture in the **Front** viewport. Double click the viewport title to maximize the view.

Scale and position the background bitmaps

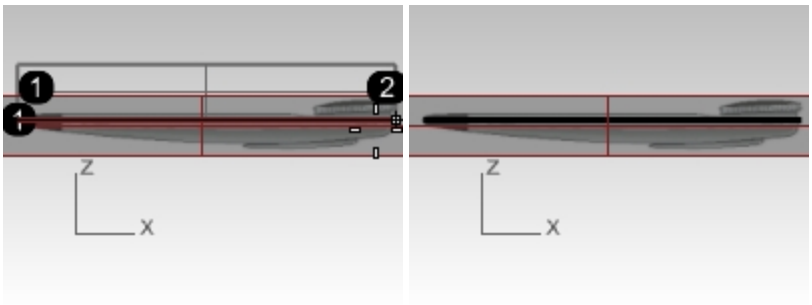
The bitmap was placed but the scale is not correct. You will orient and scale the image at the same time.

1. From the **Transform** menu, under **Orient**, click **2 Points**.
2. On the command-line, set the **Orient** options **Copy=No** and **Scale=3D**.
3. At the **Reference point 1** prompt, pick at the left end of the image, but do not pick at the end of the Picture surface. (Use the ALT key to temporarily disable the Osnaps.)
4. At the **Reference point 2** prompt, pick at the right end of the image. but do not pick at the end of the Picture

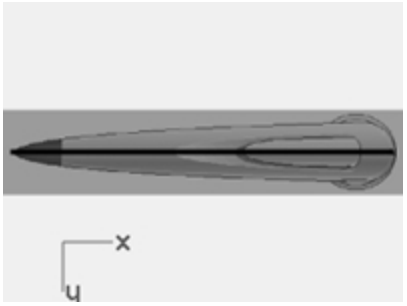
surface. (Use the ALT key to temporarily disable the Osnaps.)



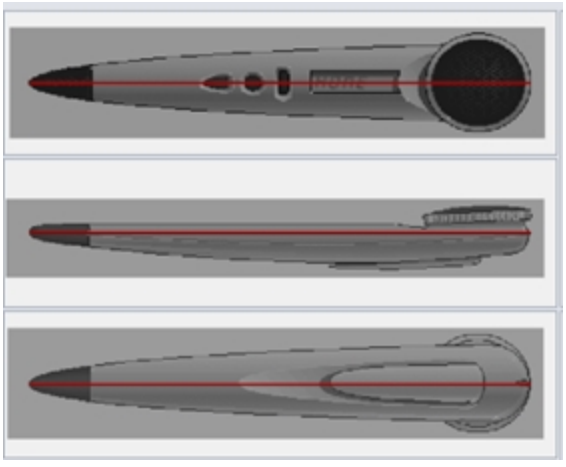
5. At the **Target point 1** and **Target point 2** snap to the end points of the 150 mm line that correspond the reference points.



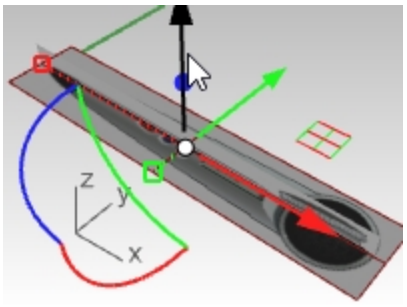
6. Change to the **Bottom** viewport.
7. Use the same technique to place and align the **HandsetBottom.bmp** in the **Bottom** viewport.



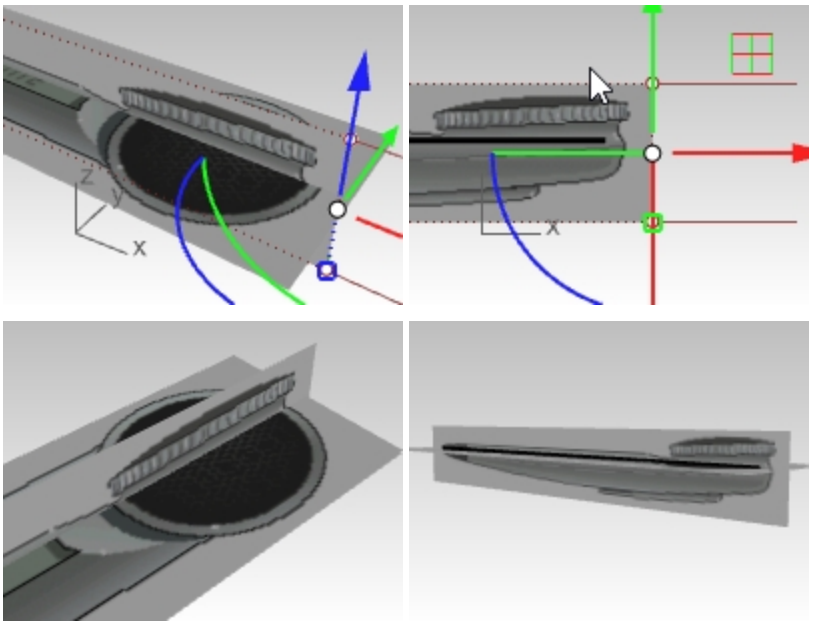
8. Repeat these steps for placement and orientation of the **HandsetTop.bmp** in the **Top** viewport.



9. Using the Gumball, move the Top Picture object by a small amount in the Z direction (blue arrow) until the top image is visible.



10. Use the Control points on the edge of the Picture object to adjust the size until all 3 images look appropriately aligned in the **Perspective** viewport.



Managing the images

Now that the images are placed and scaled, you can start using them to reference while drawing curves.

There may be times when you want to hide the objects, you probably will want to lock them so they cannot be moved by mistake. To avoid their being selectable, you can lock the layer.

You may also want to make the Pictures transparent since they may occlude objects in the 3D space.

In short you may want some tools to manage the things - this becomes more important in more complex models with multiple picture objects.

Layers

We placed the image planes on their own layer called **Bitmaps**. This layer can be turned on and off to show and hide the objects that are on that layer. The layer can also be locked to keep the objects from being selected.

Snapping to the Picture

One more step can make modeling a little easier as well. You probably do not want object snaps to snap to the Picture objects. You can control this, assuming the layer is locked, using the **SnapToLocked** command.

By disabling this command, and locking the layer the Pictures are on, you will not snap to these Picture objects. This command also has a Toggle option and it is nestable - you can use this command any time to set snapping behavior for locked objects, even inside another command.

You can make this command easy to access by assigning this macro to a Keyboard shortcut In Options.

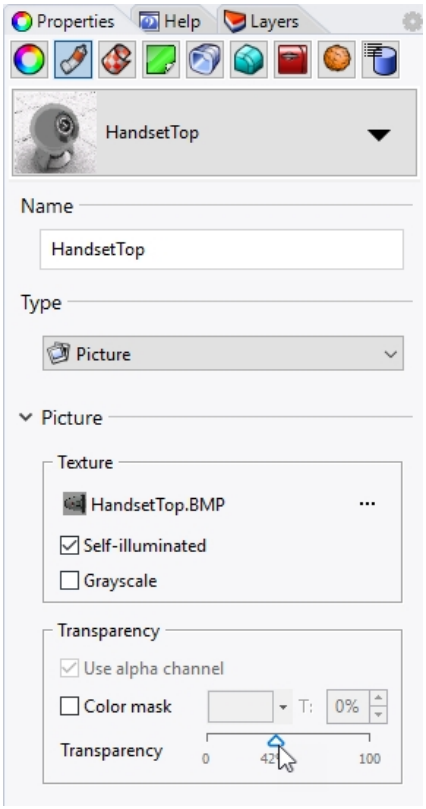
```
'_Snaptolocked _toggle
```

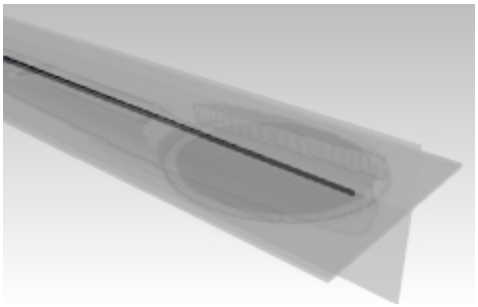
Transparency

If the colors are too saturated or you want to dim the reference images or make them transparent to a degree in order to see other geometry, you can make them semi-transparent.

To do this highlight the Picture.

On the **Properties** panel, on the Materials page, in the Transparency area, drag the slider to 60-70 percent or a setting that looks appropriate on your system. Set transparency so that the image is just clear enough to trace against but not completely occluding the rest of the scene.





Drawing the curves with InterpCrv

Keep in mind the previous discussion on layer visibility, locking and other 'management' tools and making changes as needed as we go. Next you will start by drawing the curves.

You will be creating curves to define the basic shape of the case, but not any of the details or other features. For additional practice, work to finish the model beyond the details provided in this text.

There are two curve commands that are obvious choices to make the curves.

The first is the **InterpCrv** command, while the second is the with the **Curve** command.

While the more obvious choice for tracing is typically the **InterpCrv** command since you can pick exactly, along the pixels in the images, where you want the curve to fall. It is not always the best tool.

Drawing the curves with Curve

Another way to draw these curves is to use the command **Curve,** also called the Control Points curve.

1. Place the curve control points approximately where they should to be located to define the shape and use approximately the correct quantity of control points.
2. Next edit the curve to further refine the shape.
3. To refine the shape, add or remove control points as required.

With this method, creating the curves can accurately match the image with fewer control points and cleaner curves.

Authorized Rhino Trainer Gary Dawson, of Gary Dawson Designs, covers this concept in this video:

Next you will draw four curves in order to define the required shape:

- one in the **Bottom** view, representing one half of the shape in that view
- three in the **Front** viewport: top edge or outline, the bottom edge and the curve in the middle which is the parting line curve.

The most useful tool for tracing free-form curves is a control point curve or the **Curve** command. Here are a few pointers for the next part of this exercise.

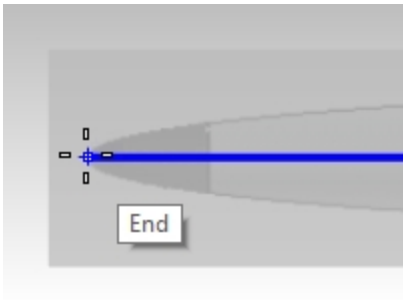
- Place the fewest number of points that will accurately describe the curve.
- Avoid the trap of trying to be 100% accurate with every point placement.
- With some experience you will be able to place about the correct number of points in about the correct places.
- Then use control point editing to adjust the curve into its final shape.

In this example, the 2-D curves can all be drawn quite accurately with a degree-3 curve using five or at most six control points.

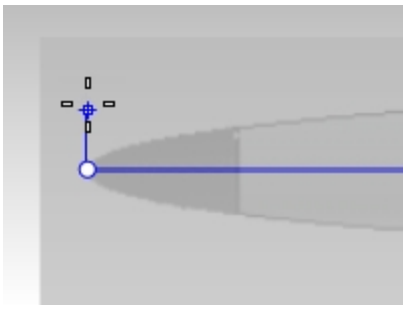
Remember to pay attention to the placement of the second points of the curves to maintain tangency across the pointed end of the object.

Drawing the Bottom view curve

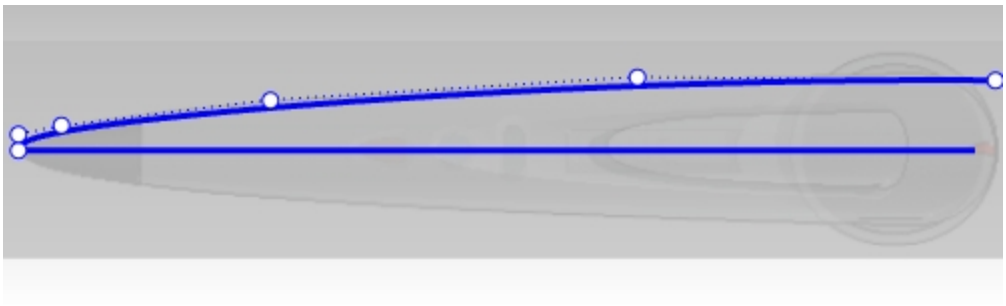
1. From the **Curve** menu, from Free-Form, click the **Control Points** command.
2. In the **Bottom** viewport, set the start of the curve by snapping to the end of the reference line
   All the curves will start by snapping to this same curve end point.



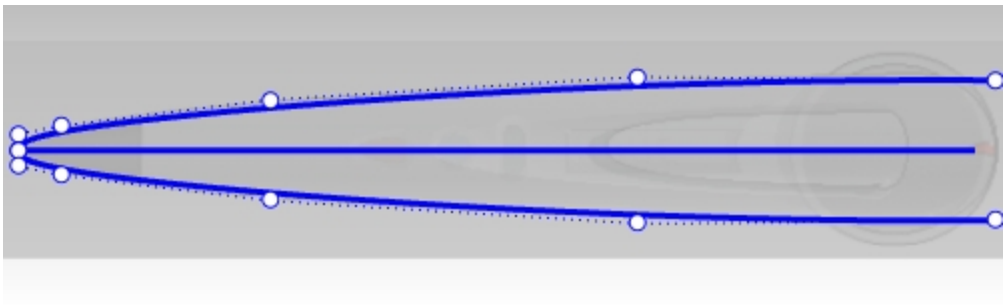
3. Click the second point directly above the end point in the CPlane Y direction using Ortho or SmartTrack. This will ensure that a mirrored copy of this curve will have good continuity across the end point.



4. Place four more points - six total should be plenty, five may be enough.
   Overshoot the open end of the shape - we'll be trimming that off anyway shortly.
5. Adjust the points to make the curve match the image closely - notice that with so few points, the simple shape is very easy to match up. Gumball is a great tool for making these adjustments.



6. Use the **Mirror** command to mirror the curve over the reference line.

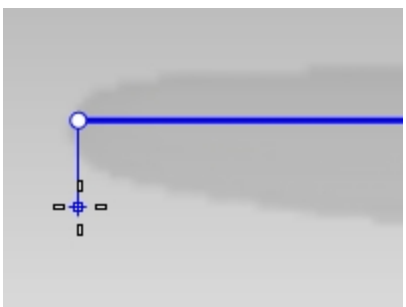


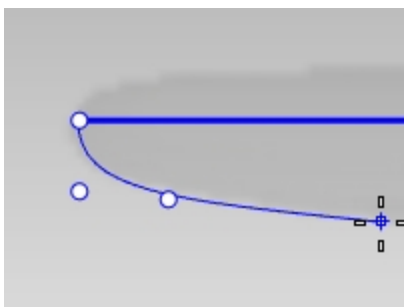
Tips for Tracing Reference Images:

- Generally areas with more curvature change will have more closely spaced points.
- Make sure that any adjustments to the second point from the tip of the shape is constrained to the Y axis direction.
- This will ensure that the curve tangent direction stays aligned to the Y axis.

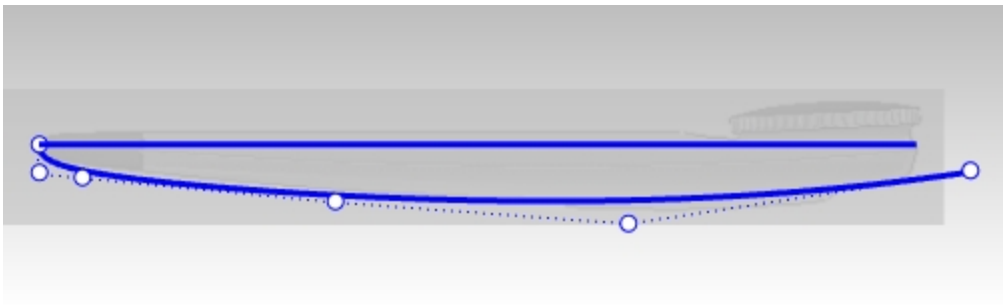
Drawing the Front view curves

1. Set the second point directly below the end using Ortho- again, this sets the tangent direction to be parallel to the CPlane Y axis.

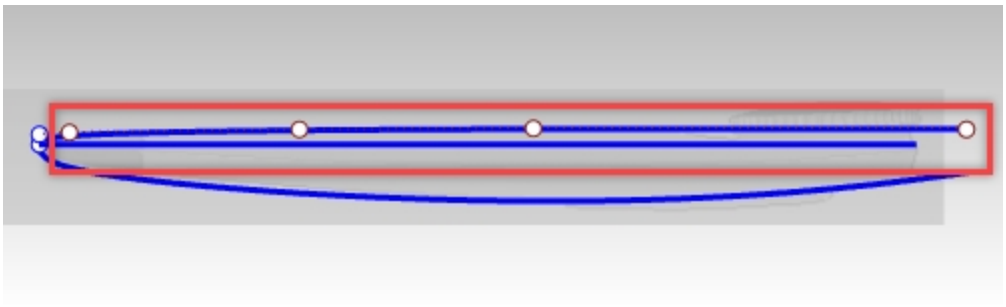




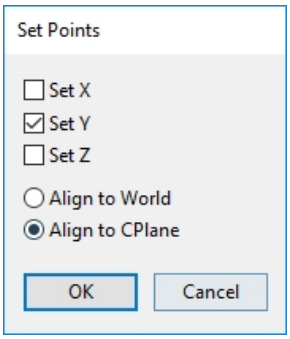
2. Place the rest of the points. Don't worry too much about how far down the point falls, They will be adjusted later.

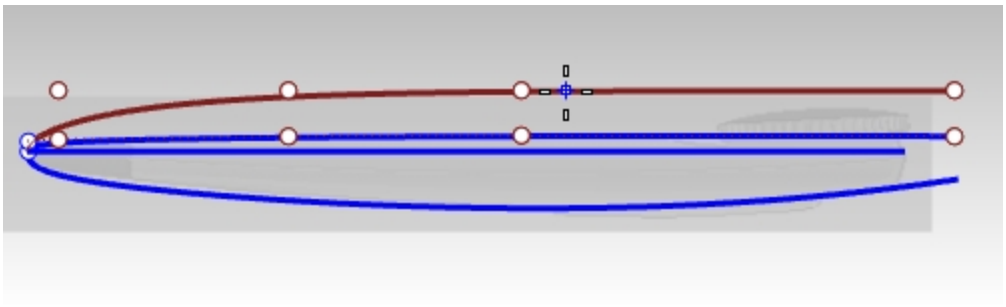


3. The top edge - this is almost exactly the same process as the previous curve.
4. To make sure the points along the straight part of the outline are lined up horizontally, use **SetPt** command after curve has been created.
   Window select the points on the top of the curve, but avoid the first two points on the curve.

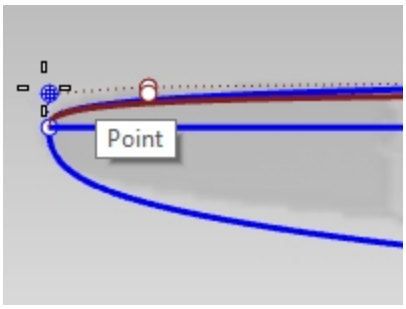


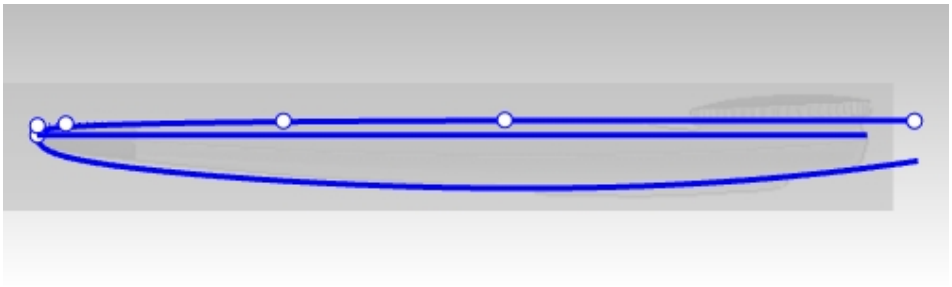
5. On the **Transform** menu, click **Set X Y Z Coordiante**.
6. Click **Set Y** for the CPlane Y coordinates only using the **Align to Cplane** option. Click the Ok.





7. At the Location for Points, pick the second point on the curve.

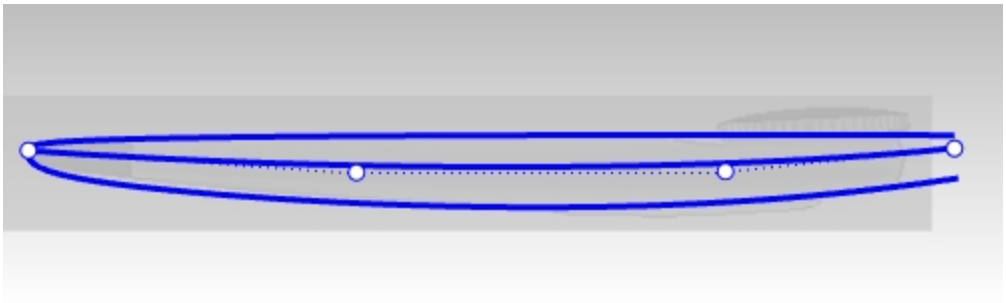




The points along the straight part of the outline are truly lined up horizontally

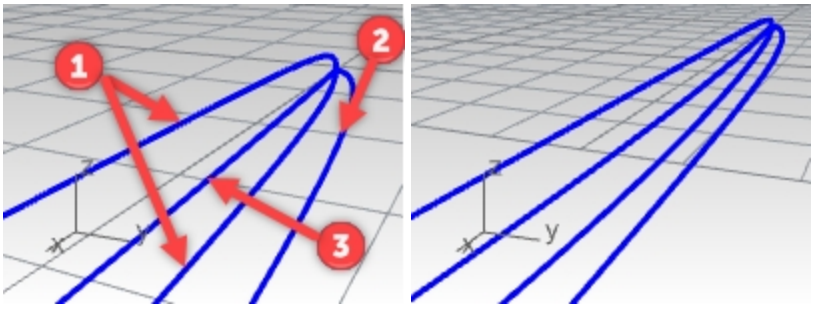
The parting line curve

1. Draw one more control point curve along the faint parting line in the image, between the top and bottom curves. This one only needs four points.



Option: Draw a line curve by picking two points. Use the **Rebuild** command to rebuild the curve as a *degree-3* curve with *four control points*. Control point edit as required to match the parting line.

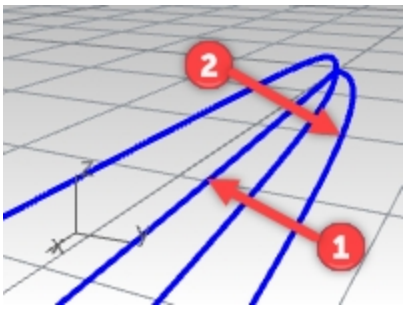
2. You should now have four curves. Turn off the reference layers to hide the pictures in order to see the curves clearly. (1) Curves in **Front** view, (2) Curve in **Bottom** view, (3) Parting line curve in **Front** view

3. 

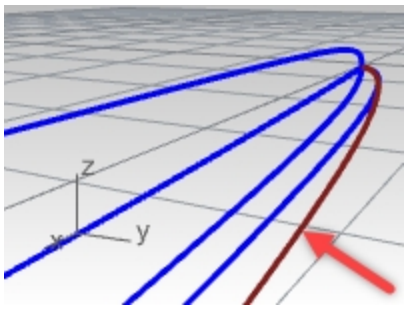
Make a 3d curve from 2d curves

All of these curves are planar and two of them actually represent 2d views of the same 3d curve. The curve you drew in the **Bottom** view is a trace of the same part of the object as the parting line curve we drew in the elevation view. You need to build the curves that represent where the curves fall in 3-D space.

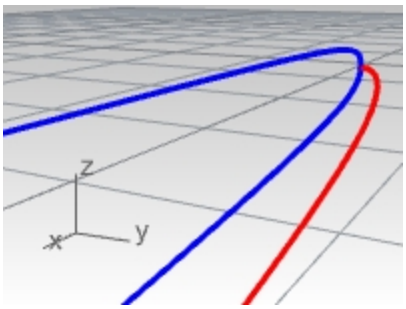
1. In the **Perspective** viewport, select the parting line curve (1) and the outline curve that you created in the **Bottom** viewport (2).



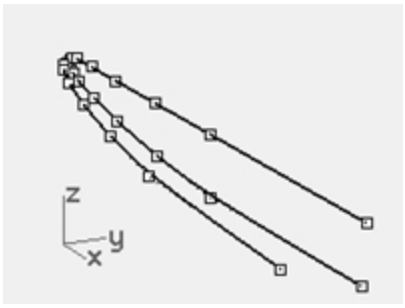
2. Use the **Crv2View** command *(Curve menu: Curve From 2 Views)* to create a curve based on the selected curves. A 3-D curve is created.

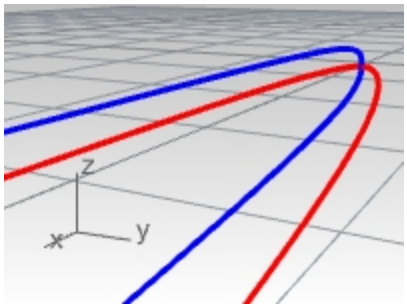


3. **Hide** or **Lock** the two original curves.
Now there are three curves.



4. **Mirror** the 3-D curve for the other side.
The macros **! Mirror 0 1,0,0** and **! Mirror 0 0,1,0** are very useful for accomplishing this quickly if they are assigned to a command alias and if the geometry is symmetrical about the x- or y-axis.

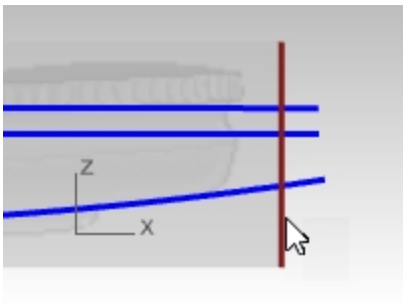




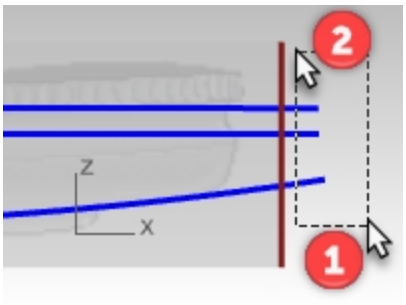
Trim the curves

Before making the surface from the curves, trim these curves in the **Front** viewport. They were drawn to arbitrary lengths. This will insure the appropriate length and shape. It will be much easier to make a clean surface if they are all cut to the same line.

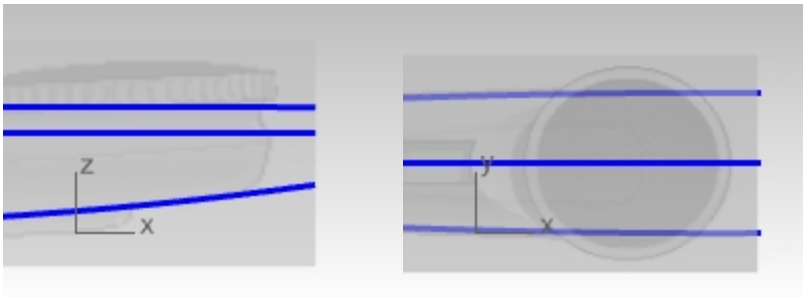
1. Turn **On** the **Bitmap** layer. The Pictures need to be visible but the layers should be locked.
2. In the **Front** viewport start the **Trim** command and pick the **Line** option. Set the line vertically between the ends of the shortest curve and the end of the object in the image. Then Enter.



3. In the **Front** view make a right to left 'crossing' selection on the curve ends



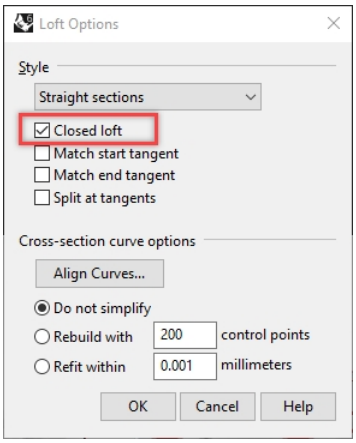
If the ends of the curves in the middle are not trimmed off, make sure the **Trim** command-line option for **ApparentIntersections** is set to **Yes** and then select the same curve ends once again.

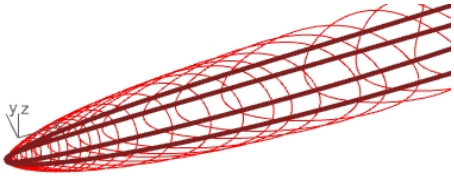


Building the surface

Often there is more than one surfacing tool that will do the job - you will see a couple surface options and decide on the one what works best.

1. Start the **Loft** command and then select the curves in order. Make sure the Loft dialog is set to create a closed loft.

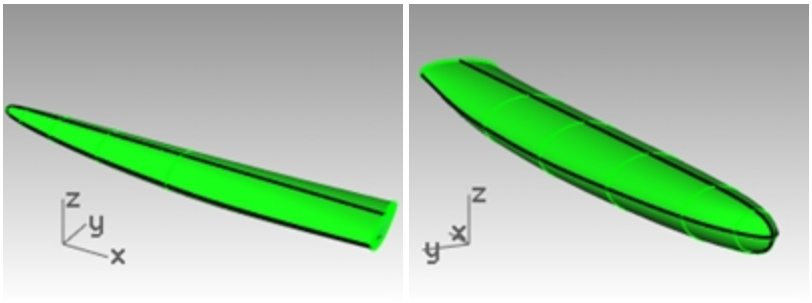




If the dialog is also set for '**Do not simplify**' the preview of the lofted surface will show badly skewed isocurves. This is a consequence of the curves being very different in structure.
In particular the 3-D curves generated by **Crv2View** are much more complex than the curves were originally drew

and these are not mixing well with the other curves in the loft.

2. **Loft** the faired curves with the **Closed loft** option checked.
   Notice the quality of the surface and how few isocurves there are.
   A closed loft will have a seam.



Loft with the Rebuild option

One way to solve this is to rebuild and fair the 3d curves to a simple structure that matches the other curves, which are simpler curves. This is fine and should work- but it may be somewhat time consuming.

There is a short cut that produces a good surface and is quite efficient.

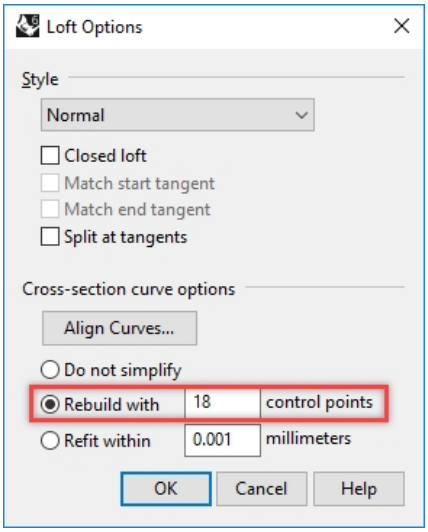
You can make a very evenly distributed smooth surface and easily adjust its points to clean up where it may not be place well, if in the Loft command you rebuild the input curves before creating the loft.

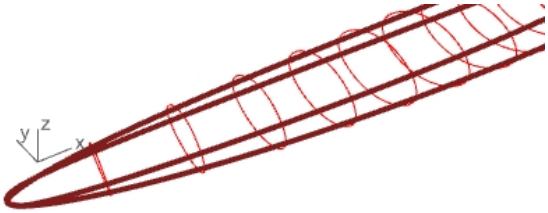
Change the setting in Loft from '**Do not simplify**' to '**Rebuild with**' and set the point count to **18**.

You will see the isocurves even themselves out nicely immediately in the preview. With this option, the input curves are rebuilt behind the scenes. Rhino makes sure that they all have exactly the same uniform structure (degree, point count and distribution of control points.)

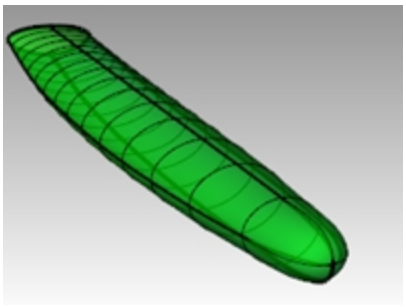
The result is pretty good, but not right at the end, where curvature changes rapidly

1. **Undo** until the previous Loft
2. **Mirror** the 3-D curve for the other side.
3. **Loft** the curves, using the **Rebuild** option to **18 points**.



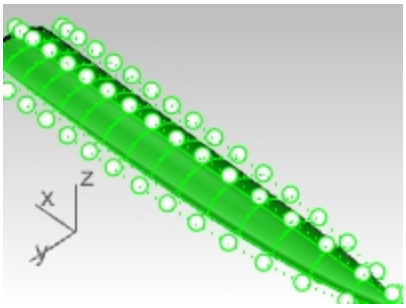


   The isocurves all true up and everything looks clean, but if you look at the tip, it falls away from the input curves.
4. The **Rebuild** option does not sample more in areas of high curvature but just divides the curves up evenly.
5. The result is pretty good, but not right at the end, where curvature changes rapidly.

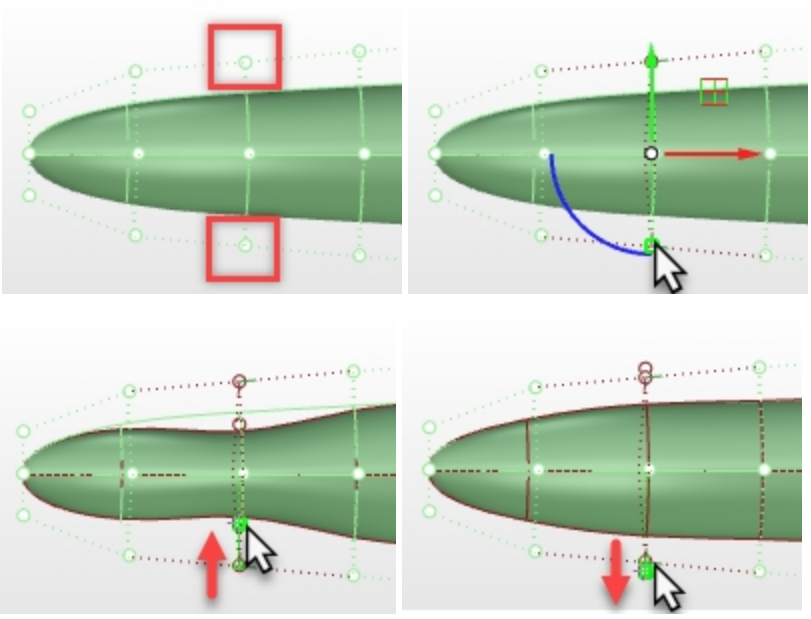


Clean up the lofted surface

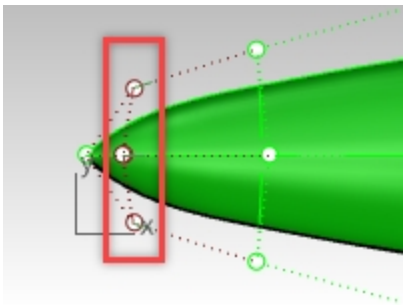
1. **Hide** the background bitmaps and set the viewport to wireframe.
2. Turn on surface points.

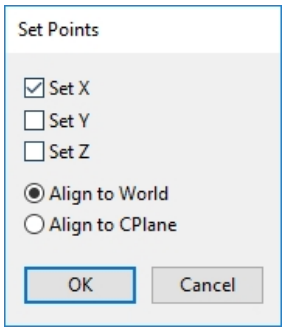


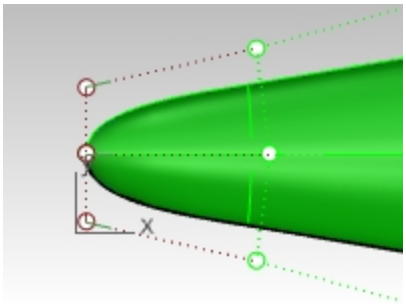
3. In the **Top** or **Bottom** viewport, move points on one side of the surface in the y-direction to get the surface to match.
   A few points moved should do it.
   Now, how do we get the other side exactly the same?
4. **Undo** all the prior moves. Next instead of moving the points, select opposite pairs of points and scale in the Y direction only using the Gumball.
   Gumball automatically uses the middle point between the pair of selected points as the origin of the scale. Turn **Ortho** on to keep the scale axis true to **Y** axis.

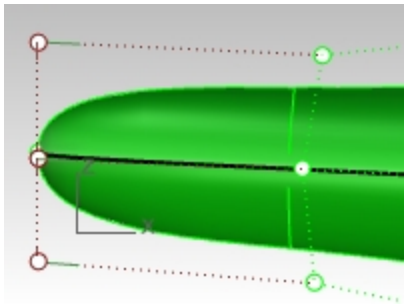


5. Use **SetPt** in the **Bottom** view to adjust the second row of control points. This will improve the location where the curvature is changing rapidly.







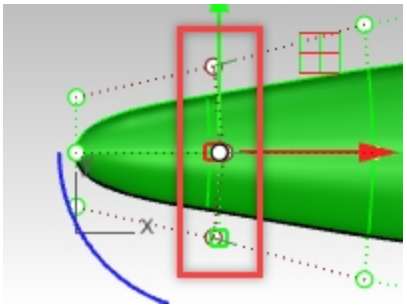


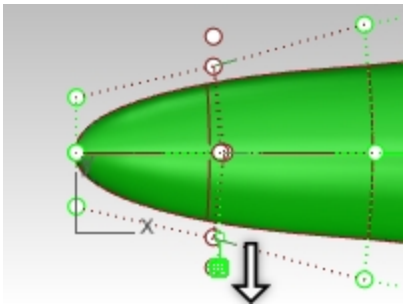
Bottom view

Front view

6. In the **Bottom** view, adjust the last two or three rings of points to make the surface match the curves there.

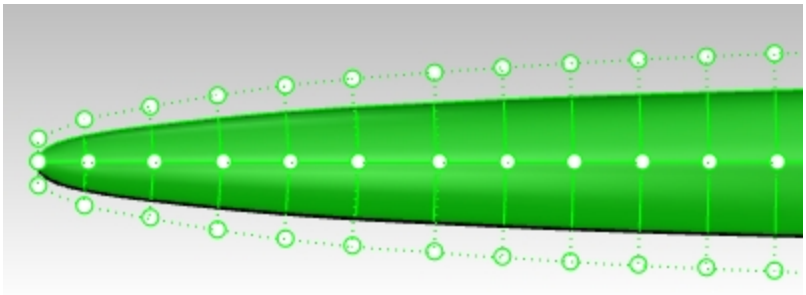
   Since the object is symmetrical in this view, it is convenient to select pairs of opposite points. Then use the **Gumball** scale handles to scale the points towards and away from each other **across** the Gumball center.

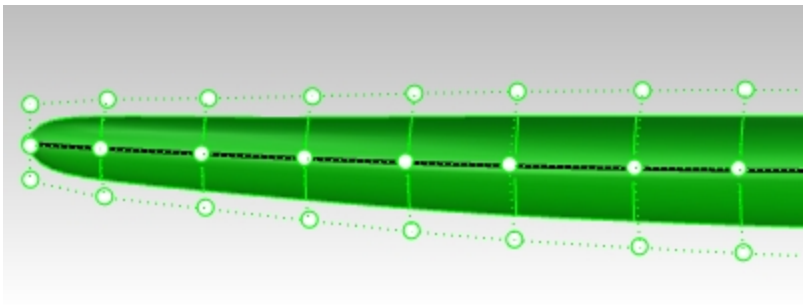


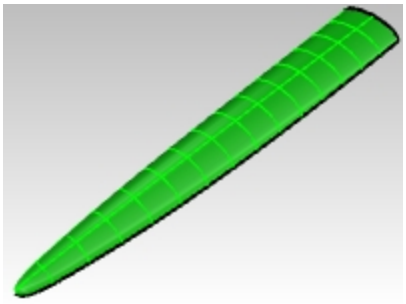


   Use the **DragStrength** command to modulate the amount of scaling - this allows fine tuning with relatively large mouse movements scaling only a little.

7. In the side view, things are not symmetrical so move individual points.



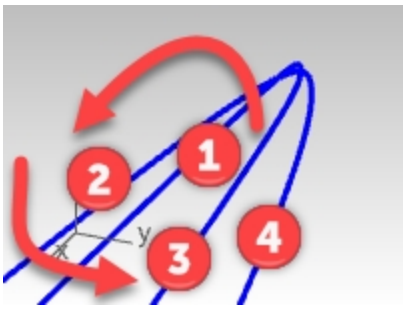


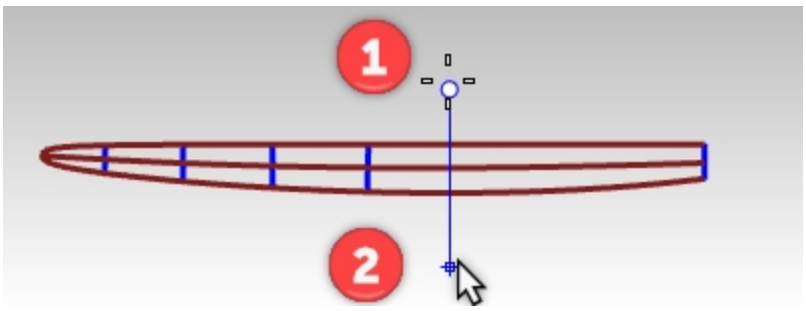


Network surface option

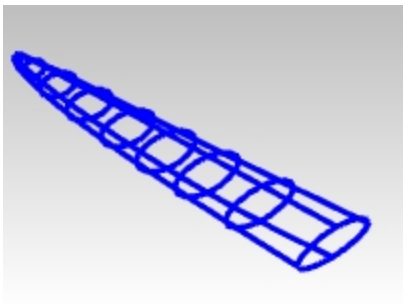
Another possibility is to create some additional curves and use the NetworkSrf command to build the surface - this command requires at least one curve in the direction 'around' this set of curves. The **CSec** command creates cross-section curves through profile curves.

1. Hide or delete the lofted surface.
2. Start the **CSec** command *(Curve menu > Cross section profiles)*.
3. Select the curves in order as if lofting them, make sure the **Closed** option is **Yes**, press Enter when done.
4. For the **Start cross-section line**, in **Front** viewport, pick a point on either side of the four curves.

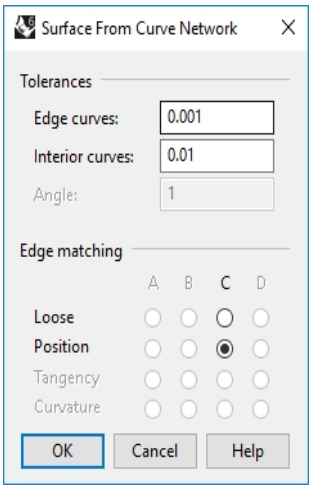


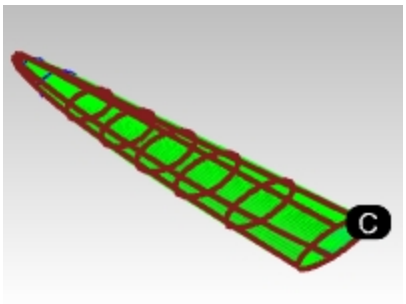


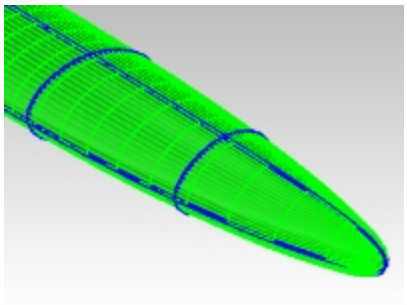
5. For the **End of cross-section line**, use **Ortho**, and pick a point on the other side of the four curves.
6. Continue this process until you have 6 to 10 cross-section curves, evenly spaced along the four curves. Make sure to add one section snapping to the endpoints at the open end of the set of curves.
   Notice in Perspective, that the result is a smooth curve interpolating between the ends of the curves. The command will keep running so you can set multiple lines in the **Front** viewport to create the smooth cross section curves. The command calculates the points of intersection of the planes represented by the lines you specify in the viewport and then interpolates a curve through those points.

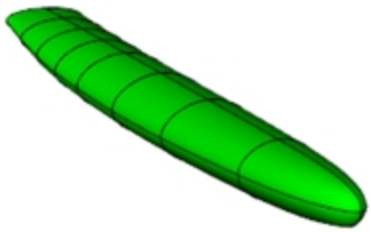


7. Window select the section curves and the original long curves.
8. Start the **NetworkSrf** command *(Surface menu > Curve Network)* to make the surface.









On your own

Using the reference images, add additional details to the handset design.

Chapter 10 - An approach to modeling

A common question Rhino users ask when presented with a complex modeling task is “Where do I start?” While there are often multiple approaches to modeling problems, start by first developing a general guideline or methodical approach. This will make modeling efficient and avoid timely remodeling tasks.

In other words, consider the shapes that need to be built and spend some time working out a strategy for laying out the surfaces that you’ll need. It is helpful to try to separate the main or ‘primary’ shapes from transition shapes or 'secondary' such as fillets and blends. A surface should not be both a primary and secondary surface, as a general rule.

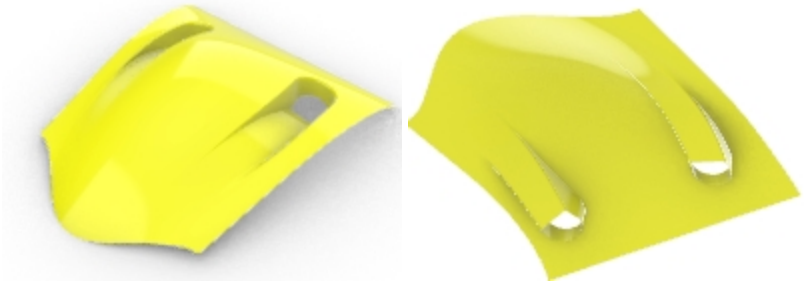
If you keep the primary surfaces as simple and clean as possible, maintaining surface fairness is much easier and adding transition surfaces can be a simpler task as well.

The Cutout

This type of advanced surface models is not limited to any one industry. The cutout techniques can be applied to a car hood, bike helmet, a boat vent, a roof or any model that needs a cutout surface with a smooth transition back to the main surface.

Exercise 10-1 Set up and build from the "floor" surface

The modeling goal in this exercise is to build a cutout or scoop in a surface which integrates seamlessly and naturally with the surface. You will start with the existing surface and some 2-D curves that define the shape we’re shooting for - and you will build some reference curves and some simple primary surfaces that have good continuity where required. In final steps, you will work on getting the transition surfaces in place with the required continuity.



Open and prepare the model

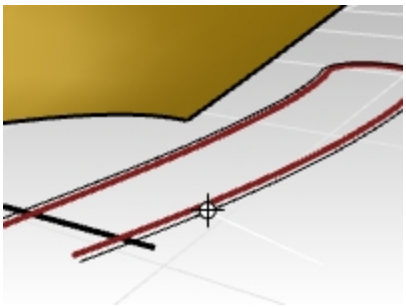
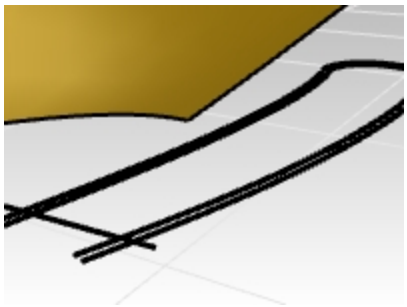
1. **Open** the model **Scoop.3dm**.
   You will see a version of the desired result as the file opens - this is what we’re shooting for.
2. Select the **Cut-out curves** layer in the layer panel, right click and choose **One layer on**.
   This will set the ‘**Cut-out curves**as current and turn off the others.
3. In the Layer panel make the following changes to the layers:

Layer	State
Cut-out curves	on and current
All other layers	on
Completed Scoop	off

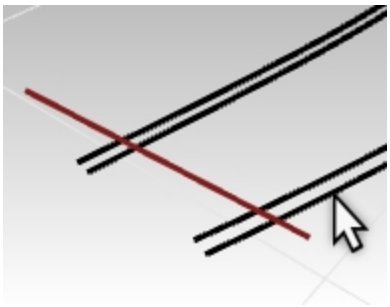
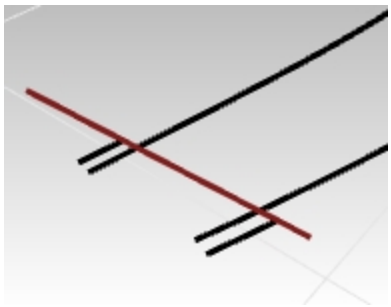
Project and extend the curves

Next, you will set up some reference curves on the surface. The floor of the scoop will be ‘overbuilt’, that is, made larger than you need it, is trimmed back to the size and shape later. This allows us to make the surfaces very simple and rectangular without being too concerned at this point about the exact shape of the edges.

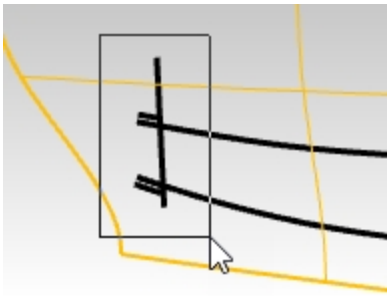
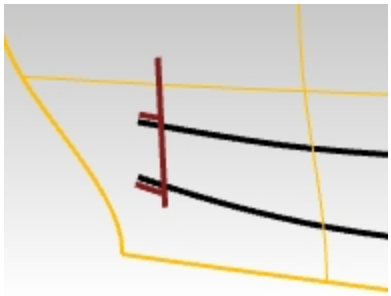
1. In the **Top** viewport, select the outside curve. The Cut-out curves layer should be on and current.
2. Click **Curve** menu, Offset and Offset Curve.
3. Set the Distance to 5 and offset the curve to the outside.

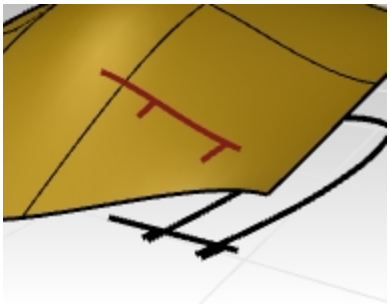
4. Use the line to trim off the loop part of the offset, leaving two short curves next to the line.

5. Select the line and the trimmed curves only.

6. Start the **Project** command *(Curve menu: Curve From Objects > Project)*.
7. **Select** the surface and press Enter.
   The curves will be projected onto the surface.



   The projected curves will help us to set up the curves that define the floor of the scoop.

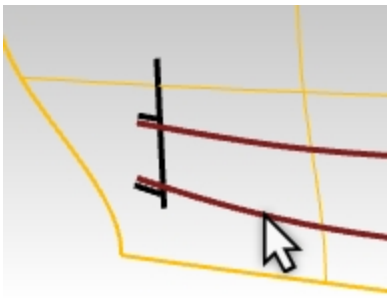
Creating the side walls

In order to create the side walls of the scoop, you can extrude the shape downward from the surface but we'll need to get that shape curve up onto the surface first. You will use the **Project** command as you did with the curves in the previous step.

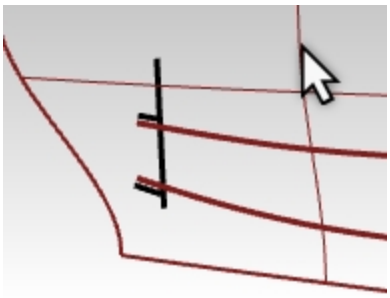
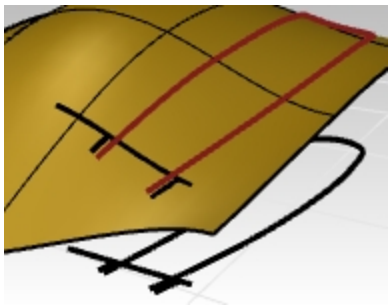
Keep in mind that the**Project** command uses the tolerance settings. (For more information on tolerances, see this page.)

Projected curves are generally more complex than the originals. It is a best to keep the curve simple as possible. To accomplish this, you will use the **Loose** option in the **Project** command.

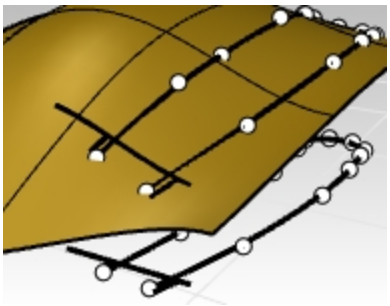
1. In the **Top** viewport, select the 2D scoop shape curve.



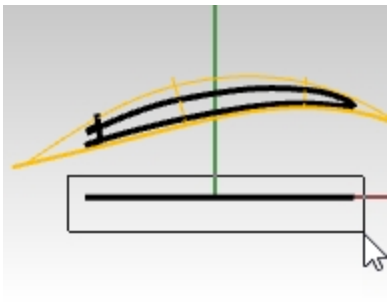
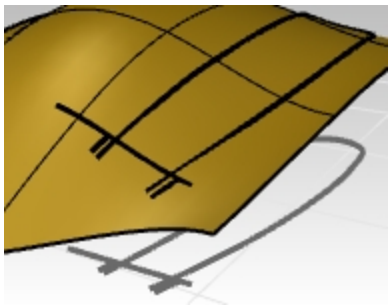
2. Start the **Project** command.*(Curve menu: Curve From Objects > Project)*.
3. Before clicking on the surface as the target, set the command-line option to **Loose**=**Yes**.
4. Next select the target surface and Enter to finish the command.

If you turn on control points for this projected curve, you will see that it has the same control point structure as the original 2D curve. Note that using the Loose option results in a curve that is not necessarily within tolerance of the target surface. In many cases, as in this one, it will be close enough and a little simpler and cleaner, which is a good thing when using the curve as an input to a surface.



5. Lock original curves on Top construction plane so they will be available for visual reference but will not be selected accidentally. This will be more easily done in the **Front** view with a window select.

Make the curves for the floor of the scoop

Next, you will make a surface for the bottom or "floor" of the cutout.

The cutout is rounded at one end, but you will build a rectangular surface and trim it to be round at one end.

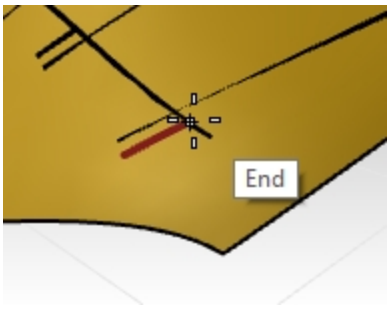
This approach allows for a much lighter, more easily controlled surface than trying hitting the edges exactly while building the surface.

First, you will make one curve with as few points as possible that best represents the shape of the part that will become the bottom of the scoop.

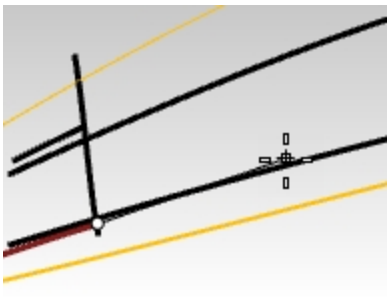
When making the curve, try to look at it from various views while you work.
Use the **Curve** command and set the command-line option **degree=5** . Use no more than six control points for a very smooth curve.

You will also check the curve with the curvature graph to get a very fair curve.

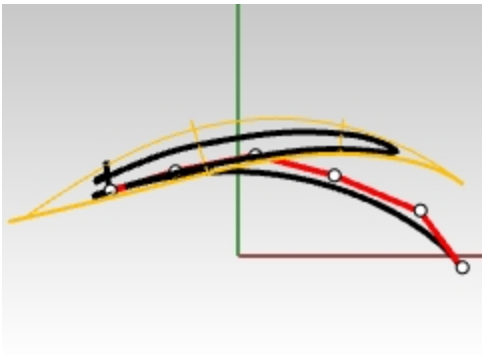
1. Open **Scoop 001.3dm** if needed. Otherwise, keep working in the scoop model from the previous section.
2. On the **Status Bar**, turn **Planar** mode on.
   This will keep the curve in a single plane for the moment.
3. With the **Front** viewport active, from the **Curve** menu, click **Freeform** and **Control Points**.
4. On the command-line, set the **Curve** command options to **Degree=5**.
5. Snap to the end of one of the short projected curves in any convenient view.



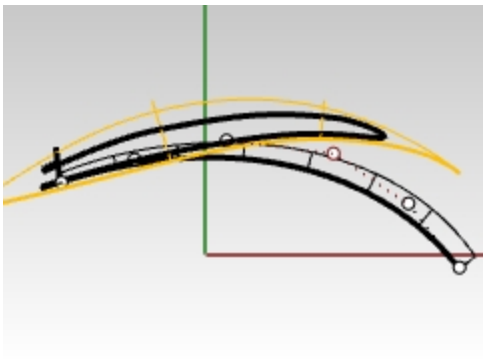
6. Set the second point just approximately in the tangent direction to the short projected curve with which you started.
   Note: You do not need to get point locations perfect the first time. You will control point edit the curve to get the exact shape soon.



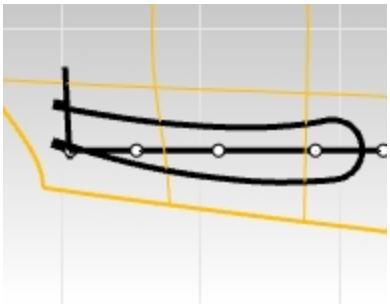
7. Switch to the **Front** viewport to continue drawing. Then place the next four points in a smooth and evenly spaced arrangement.



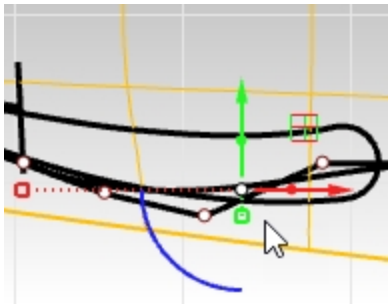
8. Adjust the control points as needed using the **CurvatureGraph** command to make a smooth progressive curve. You want the curvature to increase smoothly as the curve dives down.



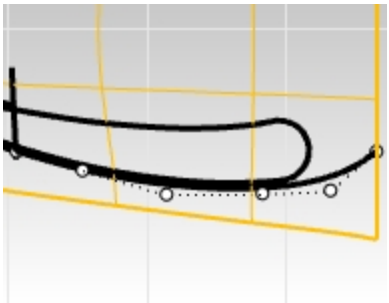
9. Adjust the curve with point editing to get the right shape in the **Top** viewport.
   Make sure to move the points only in the y-direction (Ortho will help), so that the shape in the **Front** viewport will not be altered.



   Note: The **Gumball** is recommended over dragging for moving the point. Make sure Gumball is set to align to CPlane or World (**Gumball context** menu).
10. In the **Top** view, select the last three points on the curve. Slide them using the **Gumball** green arrow, in the Y direction. This ensures that the shape as seen from Front will not be changed.



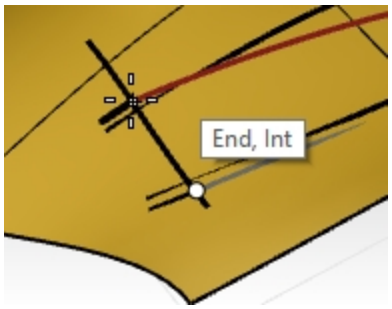
11. Make the curve roughly parallel to the scoop shape curve in the **Top** view. Edit the control points until the curve approximates the outermost of the original curves and extend somewhat past the rounded end.



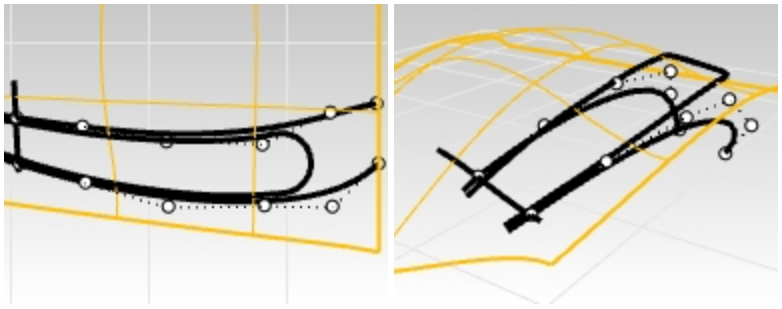
12. When this curve looks good, Copy the curve over to the end of the other short projected curve. It will need some editing but will be a good starting point for the second curve.

Copy the curve

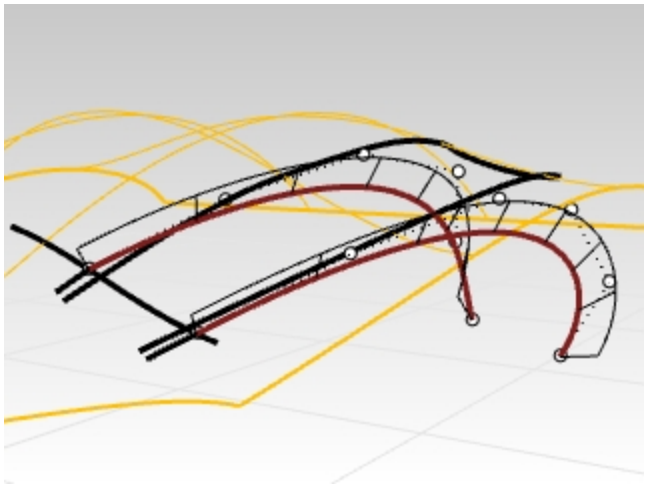
1. **Copy** the curve to the other edge.



2. Adjust the curve with point editing to get the right shape in the **Top** viewport.



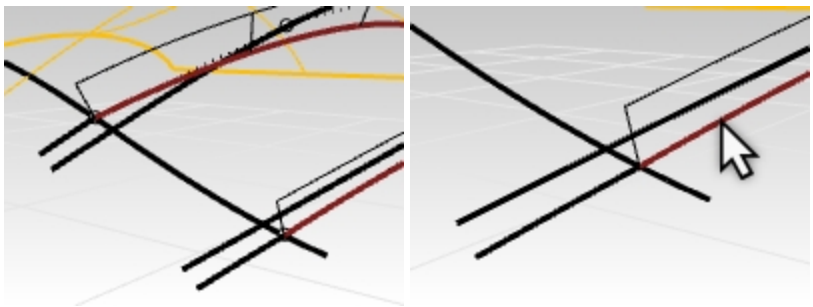
3. Use point editing, Match, CurvatureGraph and EndBulge to massage this curve in the same way as before to be clean and continuous with the projected curve, and have the same curvature character as the first curve.



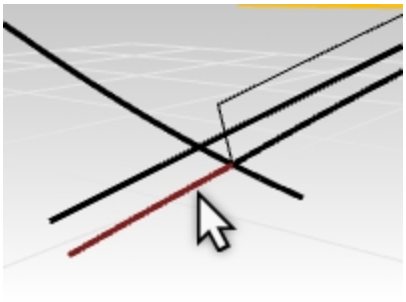
Fix the continuity

To fix the continuity, you will first make sure the new curve is matched for curvature to the little projected curve that you used as the starting point. This is really why you projected that curve onto the surface. You will match our new curve for Curvature to the projected curve which lies on the surface. This will ensure that the new curve is well aligned and matched to the surface itself. This will setup the floor surface so you can easily edit it to match to the main surface.

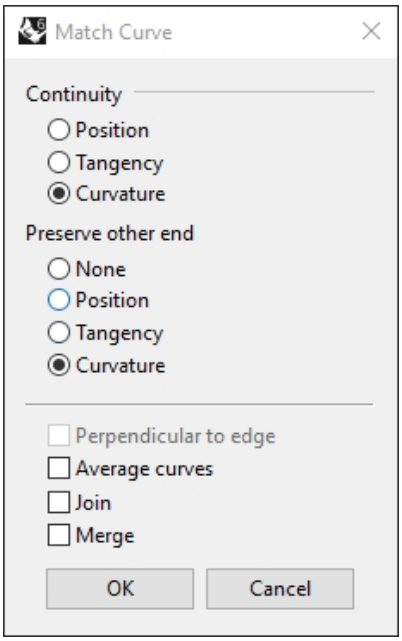
1. Use the **Match** command *(Curve menu: Curve Edit Tools > Match)*. Select the curve you just edited.



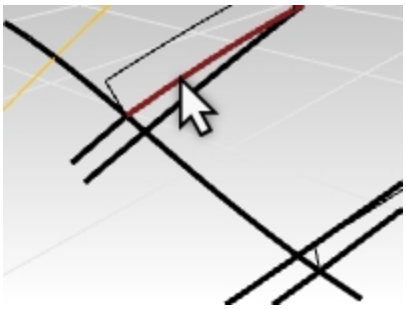
2. Next pick the trimmed curve as the match for the **Curvature** continuity.



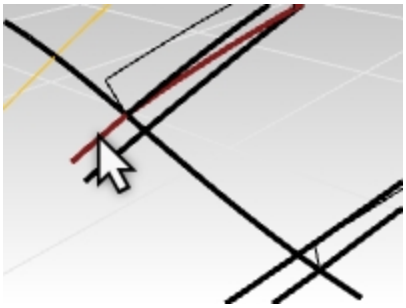
3. In the **Match** dialog:
   Continuity is set to **Curvature**
   Preserve other end does not matter in this case, but you can set to **Curvature**.
   Do not check **Average**, **Join** or **Merge**.



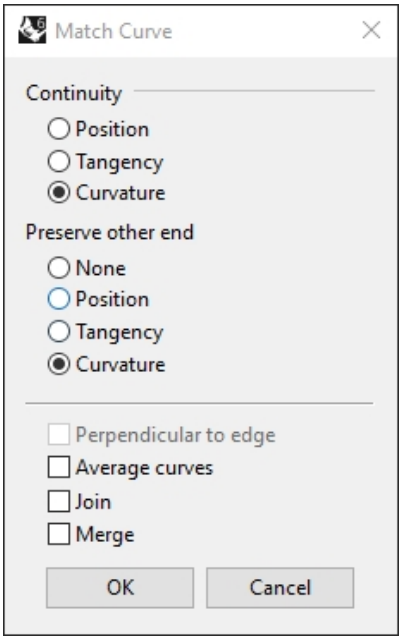
4. Next, use **Match**again to adjust the continuity of the copied curve.
   Select the curve that you just copied and edited.



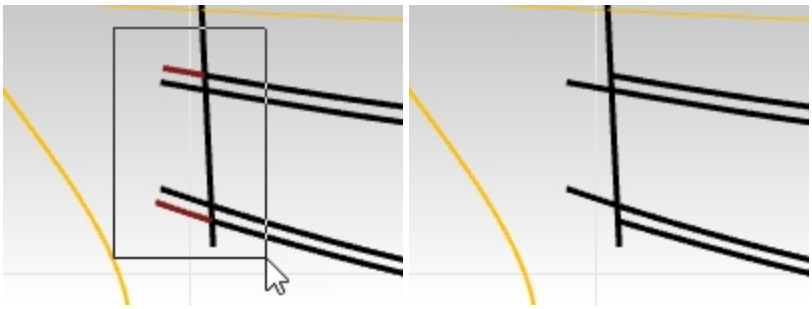
5. Next pick the trimmed curve that you copied earlier.



6. In the **Match** dialog:
   Continuity is set to **Curvature**
   Preserve other end does not matter in this case, but you can set to **Curvature**.
   Do not check **Average**, **Join** or **Merge**.



7. Use of the **EndBulge** command and further point editing may be needed.
   Make any final tweaks to the curve. The previous Match operation may have made the curvature acceleration a little messy.
   Keep in mind that editing the last three points on the curve is ok, but the end that you matched earlier is not an option if the goal is to maintain the curvature match. To edit these points use the**EndBulge** command, which constrains the point movement so that the curvature at the end is not changed. Strive for evenly spaced control points overall.
8. The short projected curves can now be hidden or deleted.
   If matching makes the curve distort too much add a knot and try again.

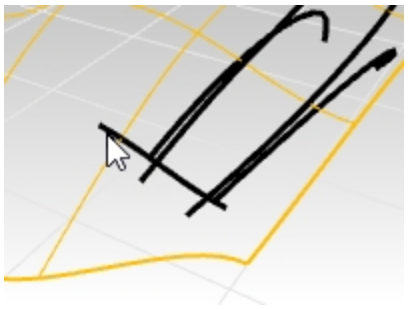


Sweep the floor surface of the scoop

Now that you have two nicely matching curves defining the overbuild for the scoop floor surface, let's see about actually creating the surface. A couple of surfacing tools come to mind immediately - Sweep2 and Loft. Either will work nicely, you will take a look at both. Sweep2 will be covered in this section.

To use Sweep2 here, you will need two rails and at least one cross section or shape curve. So far you have two nice rails only. So how can you get a cross section curve that makes sense? The projected line is a good start but it currently extends past the rail curves, you will fix that with the **SubCrv** command.

1. From the **Curve** menu, select Curve Edit Tools and SubCrv.
2. With command options set to **Copy=No** and **Mode=Shorten** , select this curve.

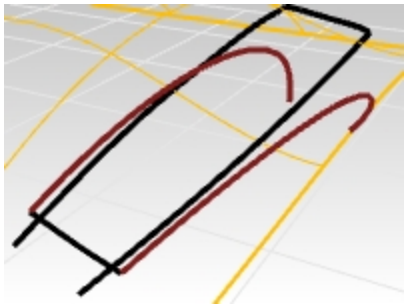


3. For the Start of curve, pick where the outside curve intersects the previously selected curve. Pick the point with the Int or End osnap where the rail curves hit the projected line to shorten the curve.

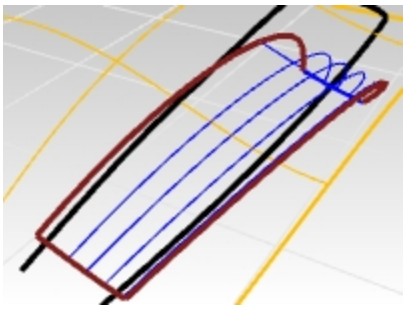
4. For the End point, select where the outside curve on the other side like intersects the previously selected curve. Pick the point with the Int or End osnap where the rail curves hit the projected line to shorten the curve.

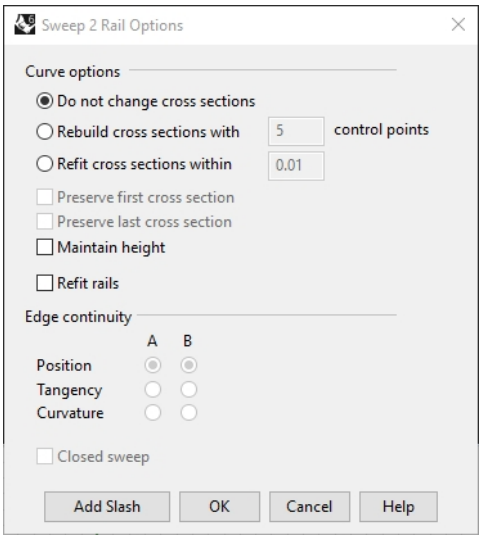
The curve is shortened to include only the portion of the curve between the two selected points on the curve. Snap to the Int or End points where the rail curves hit the projected line to shorten the curve.

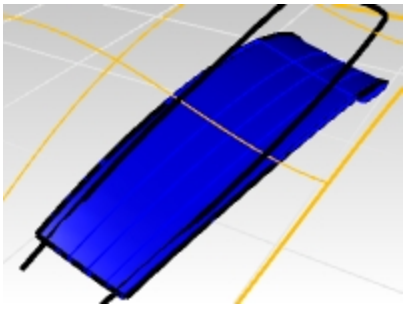
5. Select the two rail curves.



6. Start Sweep2. Sweep2 will use the two selected curves as the rails.
7. For the cross section, select the curve you just shortened as the single cross section curve.
   Note: you may want to add line between the lower ends of the rail curves and include this as a cross section curve, thus forcing the sweep2 surface to fade to a straight line at the lower end. Either way will work well.





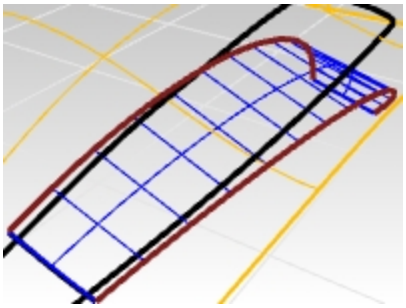


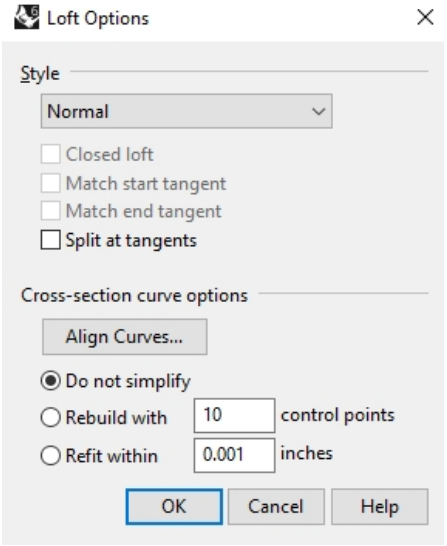
Loft a surface

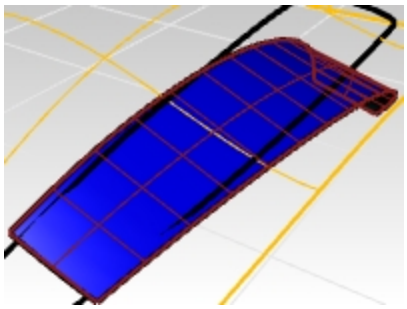
Another option is to loft the two curves to create the surface. The surface will need adjustment to match to the original surface. This will provide the opportunity to explore some options in the **MatchSrf** command

While this surface has the advantage of being extremely simple and clean, as is it has no way to attach itself to the larger surface at the front edge - it is a straight surface whereas the larger surface is curved, so there is no match, we'll need to fix that.

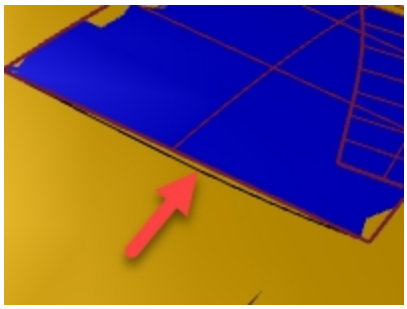
1. Select the two curves and start Loft.
2. Start the**Loft** command *(Surface menu: Loft)* to create the surface between the two curves.
   Because the lofted surface is flat, there will be a slight gap at the edge of the original surface. Make sure the loft dialog Style is set to Normal and 'Do not simplify'.







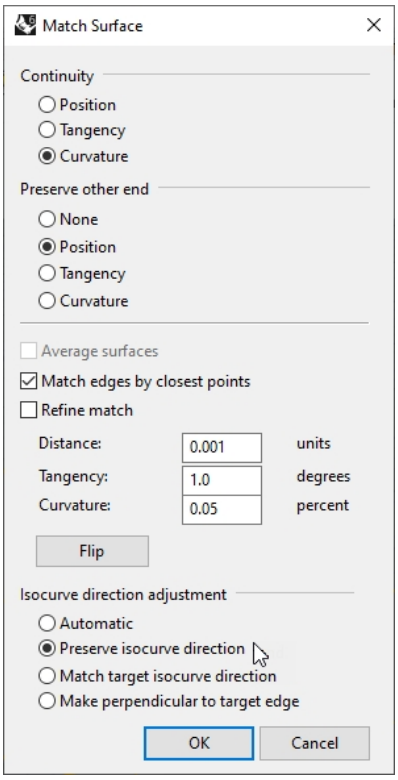
While this surface has the advantage of being extremely simple and clean, as is it has no way to attach itself to the larger surface at the front edge - it is a straight surface whereas the larger surface is curved, so there is no match. You will fix that next.

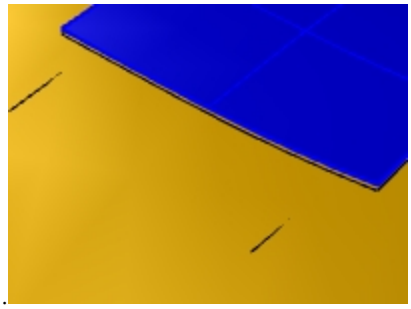


MatchSrf

Whichever surfacing tool you use, you want to be sure the new surface blends seamlessly to the larger surface.

1. Start the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)* to match the lofted surface to the edge of the original surface for curvature.
2. At the prompt to select the curve or edge to match, select the shortened line that defines the front edge of the scoop as the edge to change.
3. At the prompt to select the curve or edge to match, click the command-line option for **CurveNearSurface** to **On**. This will allow us to match to any curve or location on the target surface, not just an edge of that surface.
4. Enter and you will be promted to select the target surface. Select the larger original surface.
   The edge of the scoop floor will be pulled down to the target surface along the curve that you selected.
   The match surface with preview in the viewport.
   You may notice that the matched surface pulls around quite drastically to be perpendicular to the target edge. That will be addressed that in the next step.
5. You want to match curvature by closest points and preserve the isocurve direction of the surface.
   In the MatchSrf dialog change the **Isocurve adjustment** from **Automatic** to **Preserve isocurve direction**.
   The surface should now preview to match with much less distortion.

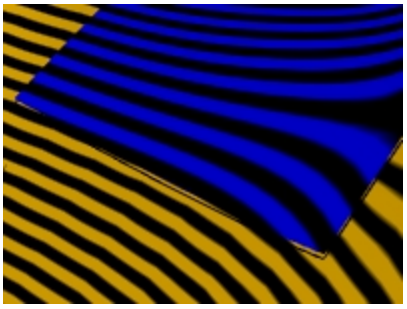




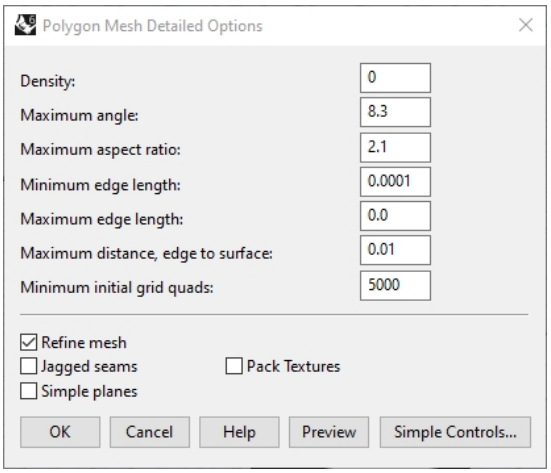
Check the continuity

To check how the previous **MatchSrf** worked, you will use the **Zebra** analysis command to visually check the continuity of the two surfaces. Because the surfaces overlap out in the middle of the target surface, you will need to set the analysis mesh for the Zebra display very fine and accurate to minimize interference between the display of the two surfaces where they come together.

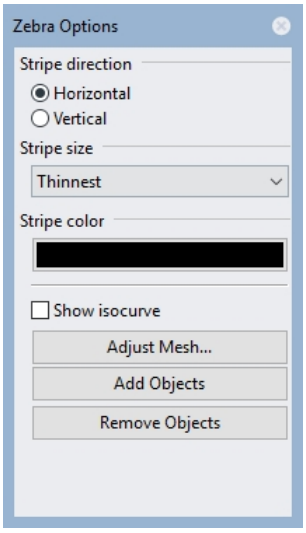
1. Select the large surface and the scoop floor surface.
2. Start the **Zebra** command *(Analyze menu: Surface > Zebra)* to check the continuity of the two surfaces.



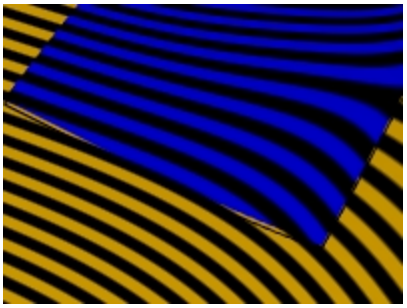
3. When the display changes to show the zebra stripes, click on the **Adjust mesh** button in the **Zebra** dialog. You will increase the density of the analysis mesh.
4. Set the **Maximum Distance Edge to Surface** to **.01**, and the **Minimum initial grid quads** to **5000**. You could even go up to 10,000. Pick the Preview button to view your new mesh.



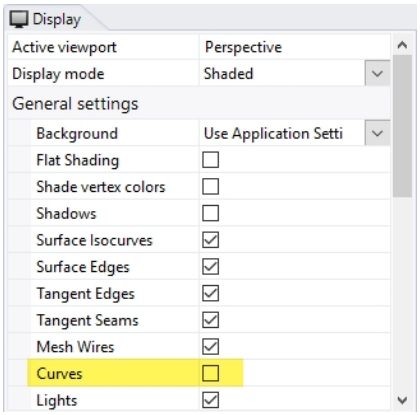
5. Set the stripe direction **Horizontal** and the **Stripe** size to **Thinner** to get a good display. Pick Ok.



6. Now you can analyze the zebra stripes with the additional smoothness on the render mesh.



Note: it may be helpful to turn off **Curve display** in the **Display mode** panel temporarily to better see the **Zebra** display at the location of the projected line.
Be careful and don't forget to turn Curves back on.

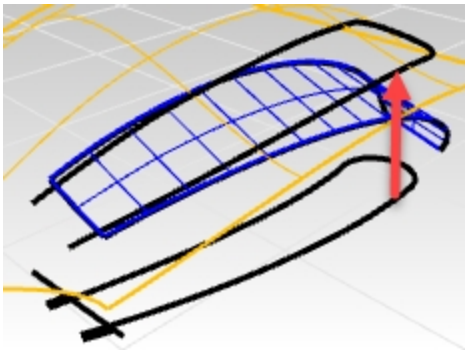


Build the sides of the cut-out

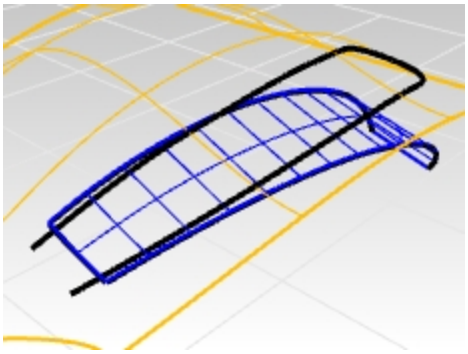
To build the sides of the cut out, you will extrude the curve downward that was projected to the original surface at the start of this exercise. You can assume the object needs to be pulled vertically from a mold, so you need to design the surface with a draft angle. You will use 10 degrees of draft and then trim the extruded surface with the lofted surface from the previous step.

Extruding the curve, tapered

1. Open Scoop 003.3dm if needed or keep working with your model.
2. If you can not located the projected curve, use the **Project** command to project the outer scope curve to the original surface.

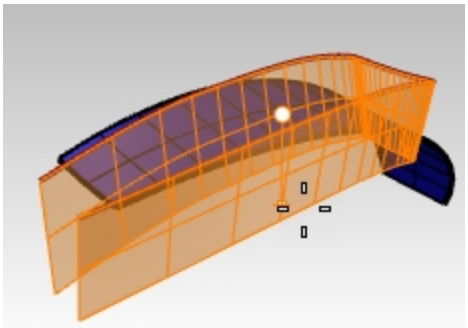
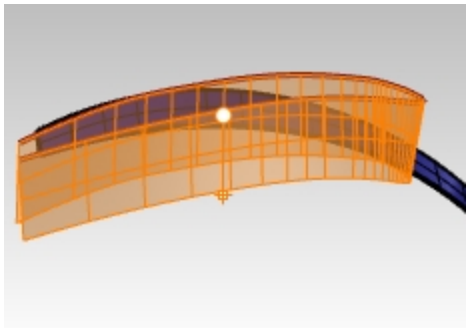


3. Using the Hide command, hide the curves on the Top cplane.

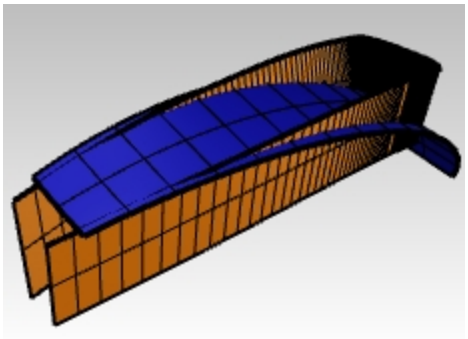


4. In the Layer panel, make the **Sidewalls** layer current.
5. **Select** the projected curve.
6. Use the **ExtrudeCrvTapered** command *(Surface menu: Extrude curve > Tapered)* to extrude the outer projected curve.
7. Set the **DraftAngle** to **10**.
   Pull the extrusion downward. With the draft angle set to **10**, the extrusion should taper inward or get narrower as you pull it down. If it gets wider instead, use the **FlipAngle** option in the **ExtrudeCrvTapered** command to change the extrude taper.

8. Pull the surface until it fully intersects with the bottom surface, stop there and pick.
   If you extrude the surface too far, you might get a polysurface instead of a single surface.
   Check this is the Properties panel.
   If this happens try to extrude again, but do not pull so far.
   If you cannot pull it far enough to penetrate the floor without making a polysurface, extrude it a short distance instead.

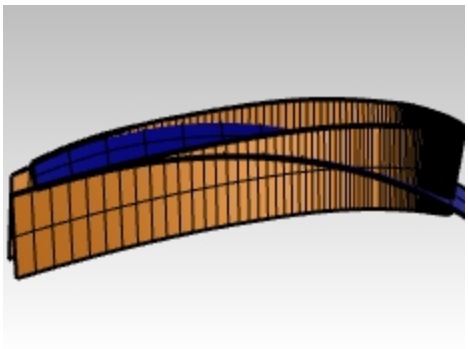
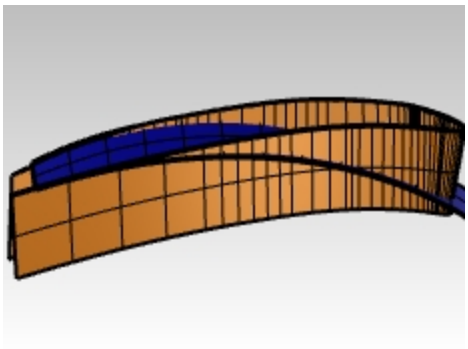



9. If necessary, use the **ExtendSrf** command *(Surface menu: Extend Surface)* to extend the sidewalls surface through the floor surface.
   The extruded sidewall surface is a very dense surface.



Refitting the surface

1. Start FitSrf and select the tapered extrusion.
2. Start the **FitSrf** command *(Surface menu: Surface Edit Tools > Refit to Tolerance)* to simplify the surface.
3. Set the command-line options to a **Fitting tolerance** of **0.001** with **DeleteInput=Yes**, **ReTrim=Yes**, **UDegree=3**, and **VDegree=3**.

Before

After

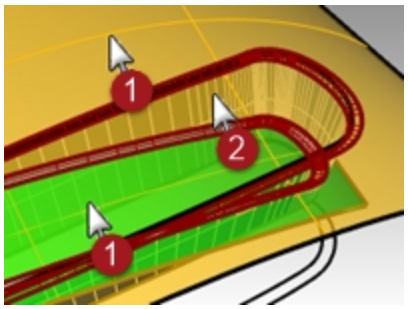
Transition surfaces

Create the fillets

First you will build constant radius 'rolling ball' fillets between the sides of the scoop and the floor surface that was built earlier and between the sides and the main surface that you started with.

1. **Show** the original surface.
2. On the **Layer** panel, make the **Fillet** layer current.
   It is helpful to be in Wireframe display mode while creating and trimming the fillets.
3. Start the **FilletSrf** command *(Surface menu: Fillet Surface)* with a **Radius=5**, **Extend=No**, and **Trim=No** to make the fillets between the bottom surface and the sides.
4. At the **First surface to fillet**, select the bottom floor surface.
5. Next, **Select** the side surface above the floor surface.

   Repeat the process for the side surfaces and the main starting surface. In this case the pick location on the large surface should be outside the scoop off to one side - this will force the fillet to roll to the outside and not to the inside of the side. The fillet surfaces will be much too long and will cross each other as well - don't worry about this, we'll trim it all up and make it nice and neat.
   Note: **FilletSrf** pays attention to where you click on the input surfaces. Since there may be up to four possible fillets between two surfaces, the pick location tells Rhino where the rails or edges of the fillets should go
   The two fillets will cross each other. You will trim them both back to their intersection points.
   Where you pick matters.



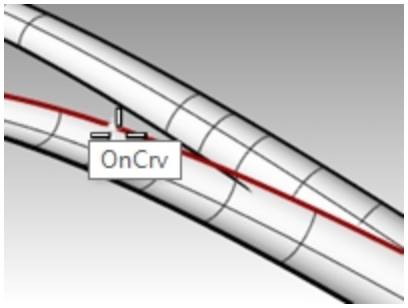
Trimming the fillet surfaces

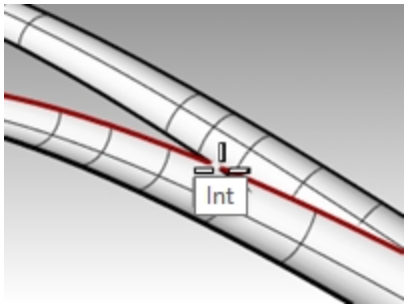
Both of the fillet surfaces are tangent to the tapered side of the scoop. Where the fillets cross they are tangent to each other.

If you trim the ends of the fillets to a plane, then the resulting trimmed edges will be tangent to each other. Trimming these surfaces will be helpful when creating the final surfaces that blend the fillets out between the scoop and main surfaces.

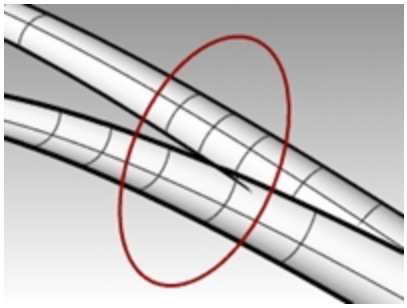
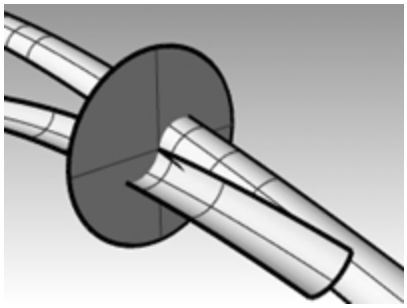
To create the plane, first make circles with the AroundCurve option around one edge of the fillet surfaces, then make planar surfaces from the circles.

1. Open Scoop 004.3dm if needed.
2. **Select** the fillets and run the **Isolate** command in the **Visibility** toolbar to isolate them.
   Every other object that is not selected and does not have control points on and is not locked, will be hidden.
3. Start the **Circle** command, and use the **AroundCurve** option. Set the **Intersect** object snap only.
   Note: Right click on **Intersect** in the Osnap bar. All the other osnaps will be turned off, and only the option you right clicked over, will be on.
   The **AroundCurve** option forces the **Circle** command to look for curves, including edge curves, to draw the circle around.

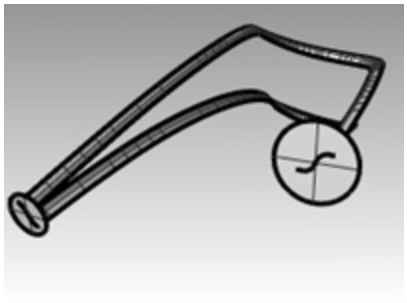




4. Click on the upper edge of the lower surface and snap to the intersection point.
5. Draw the circle out well past the width of the fillet surfaces.
6. Start the **PlanarSrf** command *(Surface menu: Planar Curves)* and pick on the circle to create a surface with the circle curve at the intersection point.
7. Repeat these steps for the other intersection.

8. Use the **Trim** command and trim off the free ends of both fillet surfaces.

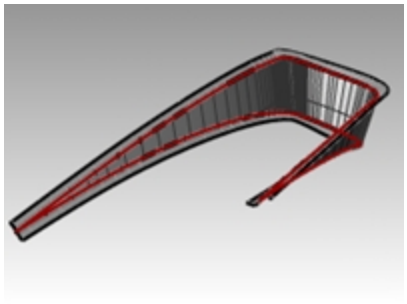


9. Repeat this for the 'collision' on the other side.
10. Delete the planar surfaces.

Trim the sides of the scoop

You can use the trimmed fillets to trim back the side surface of the scoop.

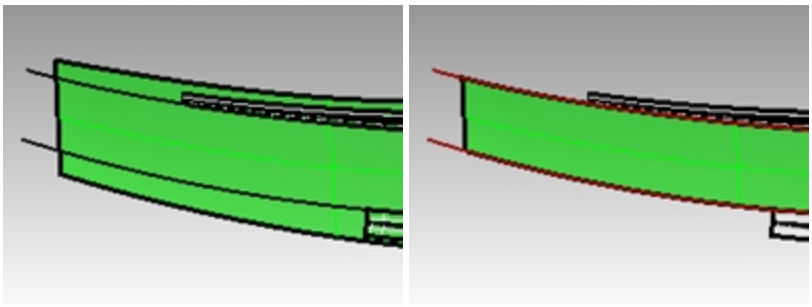
1. Use **ShowSelected** to show the tapered side surface.
2. Use the fillet surfaces as trimming objects to trim the excess from the side surface.
   It is often much faster to trim with curves than to use surfaces, especially if the surfaces are tangent to the object to be trimmed, as is the case with fillets.
3. **Duplicate** the two edges that are in contact with the side surface to use as trimming objects if you have a problem.



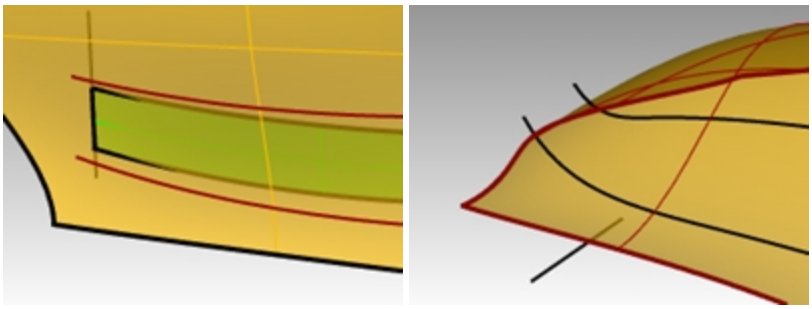
Trim the main and floor surfaces

The next task is to extend the edges of the fillets so that the main surface and the floor surface can be trimmed back. The inner, or lower, edge of the lower fillet will be extended off the end of the floor surface and the outer, or upper, edge of the upper fillet will be extended off past the end of the opening of the scoop as well. The extended curves will be projected onto the respective surfaces and used to trim them.

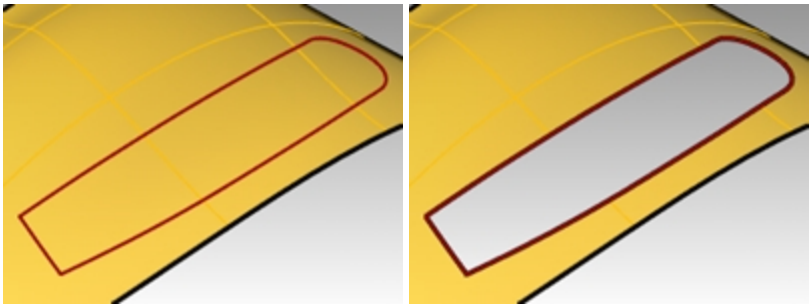
1. Open **Scoop 005.3dm**, if needed.
2. In the **Top** viewport, use the **Extend** command with the **Type=Smooth** option to extend both bottom ends of the lower fillet edge past the front of the floor surface.
3. Use these curves, still in the **Top** viewport, to Trim the outer edges from the floor surface.



4. Use **Extend** to extend the outer edges of the upper fillet past the end of the floor surface.
   Note that in the **Perspective** viewport these extended curves are off in space at their outer ends.
5. **ShowSelected** the main surface if it is hidden.
6. **Project** the curves onto the main surface from the **Top** viewport.



7. **ShowSelected** or turn on the layer for the original curves and **Project** the line segment onto the main surface.
8. **Trim** the projected curves with one another so that they form a closed loop.
9. Use the closed curves to **Trim** a hole in the main surface.



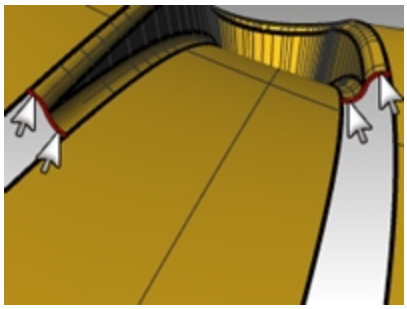
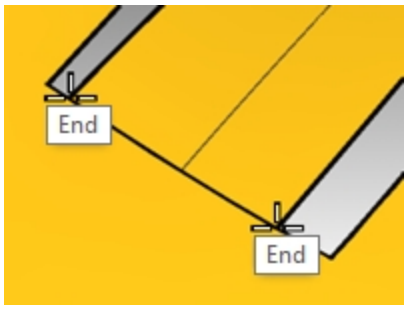
Set up the curves to create the surfaces

You are now nearly ready to create the surfaces. As you can see there are nice rectangular gaps in the surfaces, you just need to arrange the curves and edges surrounding the gaps for use in making a **2-Rail Sweep** or a **Surface from a Curve Network**.

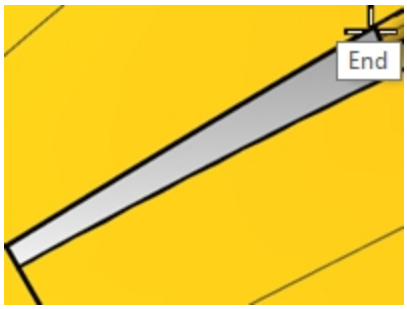
Because one end of each open rectangle is bounded by the two tangent fillet edges, you will need to create a single curve there to use as input. You will duplicate the four edges and join them into two s-shaped curves. The other end of each rectangle is bounded by a portion of the end of the hole in the main surface.

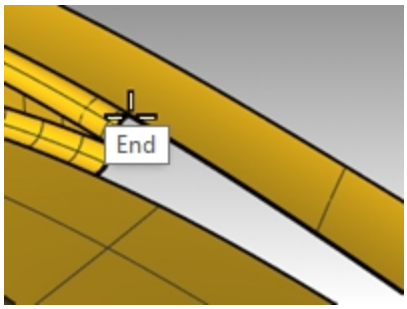
You will split up that long edge into segments that correspond exactly to the ends of the rectangular openings.

1. Open **Scoop 006.3dm**, if needed.
2. Use **DupEdge** to create curves at the trimmed edges of the fillets.
3. **Join** these four edges into two curves.
4. Use the **SplitEdge** command *(Analyze menu: Edge Tools > Split Edge)* and the **End** object snap to split the straight edge on the trimmed hole in the main surface to the end points of the floor surface's edge.

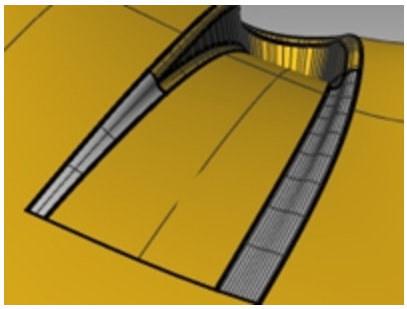



5. Use the **SplitEdge** command to split the long edges at the endpoints of the fillet edges.
   This will help **NetworkSrf** find a solution more quickly.



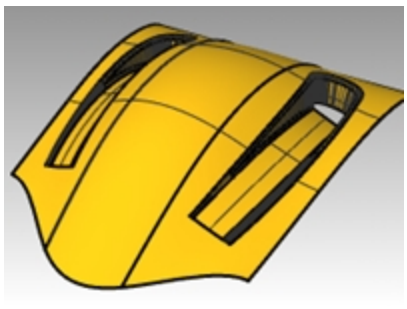
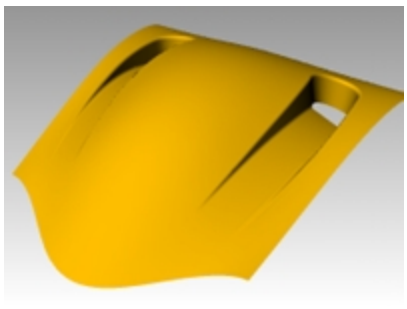


6. Use the **Sweep2** command with **Rail continuity=Tangency** or the **NetworkSrf** command to create the last two surfaces.
   The surfaces start with the s-shaped curves that you duplicated and end with a flat line at the split edges.



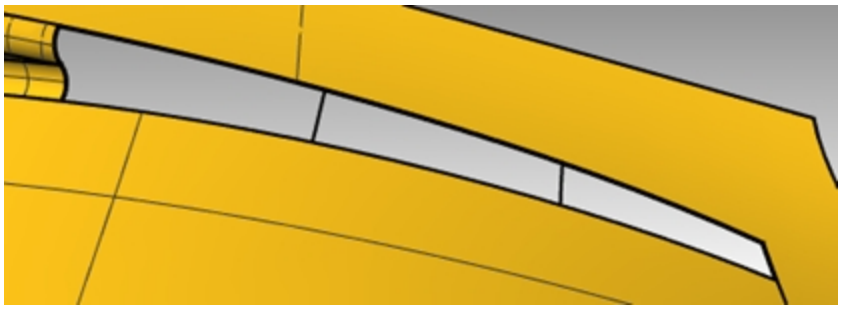
Finalize the surfaces

1. Open **Scoop 006.3dm**, if needed.
2. **Join** the cutout surfaces and then trim a hole at the bottom.
3. **Mirror** and **Trim** to get the other scoop.

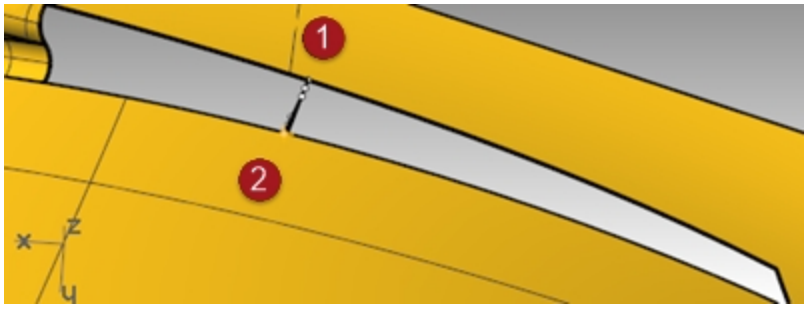



Extra cross-section curves

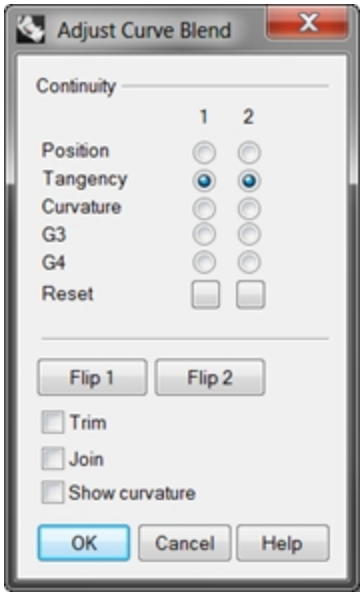
1. Open **Scoop 007.3dm**, if needed.
2. Use **Sweep2** or **NetworkSrf** to close the holes.
   Filling the larger of the two holes may benefit from extra cross section curves.
   To add cross-sections, use the **Blend** command to make tangent curves approximately one-third and two-thirds of the way along the edges of the opening. Use these curves as additional input for a network surface.



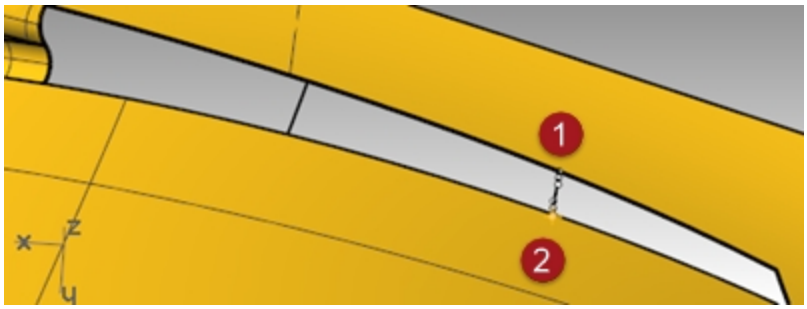
3. Start the **BlendCrv** command *(Curve menu: Blend Curves > Adjustable Curve Blend)*.
4. At the command-line, select the **Edges** option.
   Set **Continuity=Tangency**.
5. For the **Select surface edge to blend**, click approximately one-third of the way along one of the long edges of the rectangular opening.



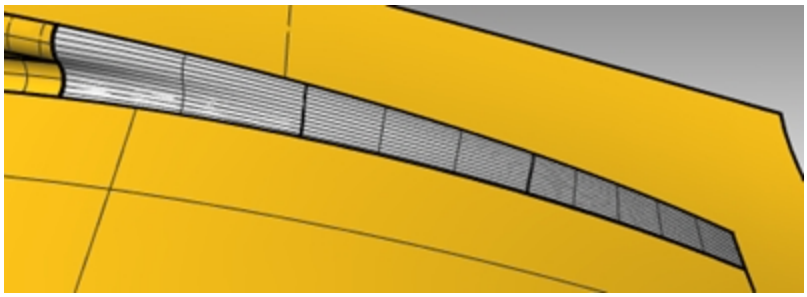
6. For the other **Select surface edge to blend**, click on the edge opposite the first one.
   The blend curve will be placed across the opening and a dialog box will open.
7. In the **Adjust Curve Blend** dialog box, set **Continuity=Tangency** for both ends.



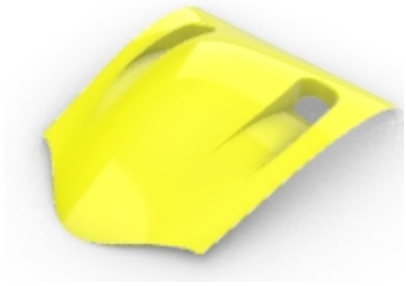
8. Make a second curve the same way about two-thirds of the way along the same edges.
   Note: If **BlendCrv** is used with **History** on, the curve can be adjusted later using the **Edit** command-line option in **BlendCrv**.



9. Use the **NetworkSrf** command to create the surface.
   Remember to include the new curves in the selection.



10. Join all the surfaces into one polysurface. Assign a material and render.



Chapter 11 - Applying 2-D graphics

Often you are asked to take an existing design from a 2-D graphics package and include it as part of a Rhino model.
In the following two exercises, we will move and position the graphic onto the model.
In this exercise, we will make a custom construction plane, import an Illustrator file, and place a logo on some surfaces.

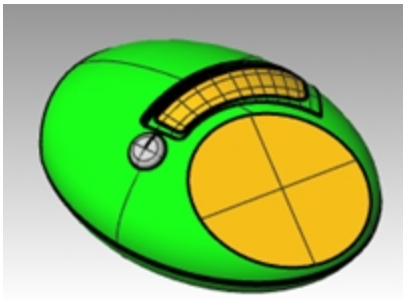
Exercise 11-1 Importing an Adobe Illustrator file

Open and setup the model

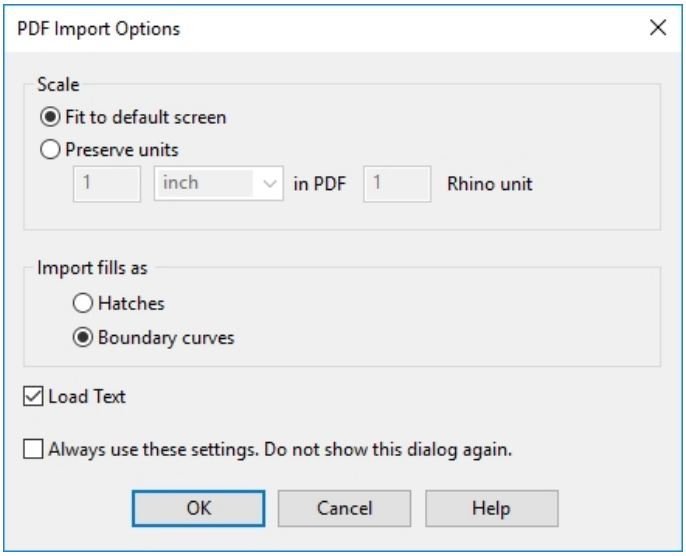
1. Open the model **Air Cleaner.3dm**.
2. In **Rhino Options**, **Modeling Aids**, set the **Construction planes** to use **Standard construction planes**.
   The following techniques will not work if the construction planes are set to use Universal construction planes.

Import a file

1. Start the **Import** command *(File menu: Import)*.
2. Change the **Files of type** to **Adobe Illustrator (*.ai)**, and choose the **AirOne_Logo.ai** to import.



3. In the **AI Import Options** dialog box, selection "Fit to default screen", click OK.
   The logo curves appear at the origin of the **Top** construction plane, are selected, and on the **Default** layer.

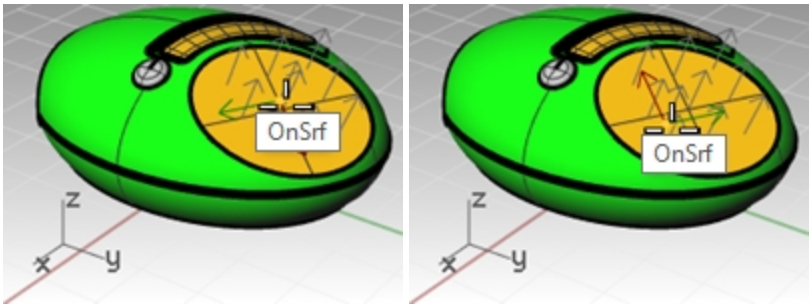


4. While the imported geometry is still selected, use the **Group** command to group the various curves together.
   This makes it much easier to select all of the curves and not leave any behind in the following transform steps.
5. Start the **Layer** command.
6. Turn off the **Logo** layer in the **Layer** panel.
7. **Right-click** on the **Logo** layer, then click **Copy Objects to Layer** to make a copy of the logo on the **Logo** layer.
   We will use this copy later for another part of the exercise.
8. Turn off all the layers except **Default, Body** and **Top Surface**.

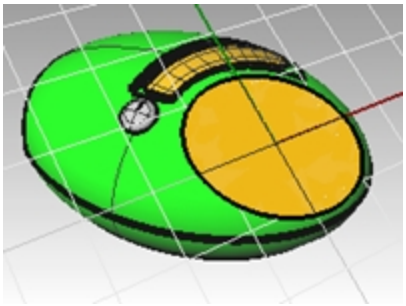
Create a custom construction plane

We need to set a construction plane to the flat surface. The **CPlane** command will allow us to do this but the X and Y directions of the new custom construction plane will be mapped to the u- and v-directions of the target surface respectively. The **Dir** command will tell you how the u- and v-direction are pointing on the surface, and allow you to change the directions of each.

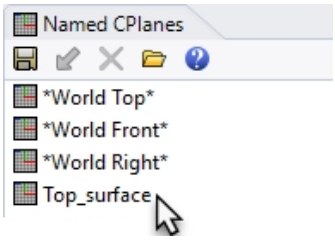
1. Set the Display mode of the **Perspective** viewport to wireframe.
2. **Select** the flat disc shaped surface, then from the **Analyze** menu, select **Direction**.
   This displays the current surface normal direction and the U/V directions. It is important to know the u- and v-directions of the surface.

   The white arrows show the surface normals. A cursor with a red and green arrow appears when you move over the selected surface.
   The red arrow indicates the U direction and the green arrow indicates the V direction.
3. At the command-line, notice the options for changing the directions of the surface. You can click on these to change the surface directions. The cursor and surface normals will update accordingly.
   When all changes are made, press Enter to accept.
   The goal is to have the U and V as in this image.

   In this way, the new construction plane will map to the surface accordingly and the geometry can be mapped to the construction plane predictably.
4. In the **Perspective** viewport, use the **CPlane** command with the **Object** option *(View menu: Set CPlane > To Object)* or *(Viewport title right-click menu: Set CPlane > To Object)* to set the construction plane to the surface.
   The x- and y-axes are parallel to the u- and v-directions of the surface as you set them in the previous step.
5. In the **Named Cplanes** panel, save the new construction plane as **Top_surface** This will make it easy to retrieve in later steps.

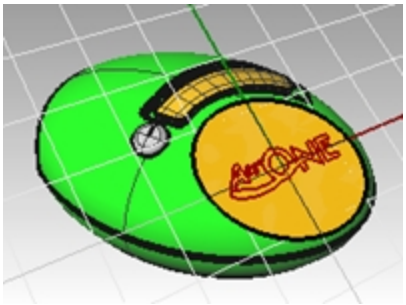


Map the logo curves to the new construction plane

The command we will use to move the logo to the flat disk shaped surface uses the position of the object relative to a construction plane.

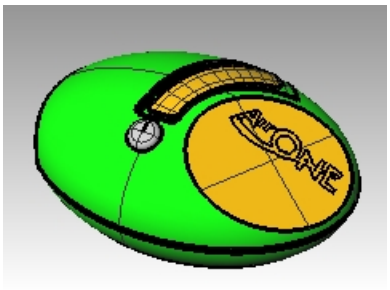
1. Make the **Top** viewport active. The Display mode of the **Top** viewport should be set to **Wireframe**.
2. Select the curves in the **Top** viewport.
3. On the **Staus Bar**, click **Record History**.
4. Select the logo curves
5. On the **Transform** menu, click **Orient** and **Remap to Cplane**.
   This starts the **RemapCPlane** command
   The **RemapCPlane** command depends upon the active construction planes at each stage, so it is important to pick in the correct viewports.

6. Click in the **Perspective** viewport with the custom **CPlane**.
   You could use the **Copy=Yes** option in this command so that a copy is remapped instead of the original.
   The logo is positioned in the same relative position on the custom construction plane as it was in the active viewport.



7. Using Gumball, **Rotate**, **Move**, or **Scale** the original logo to a new position. Since **History** was used, changes to the original (or parent) curves update the copy (or child) curves that were remapped to the custom cplane in the **Perspective** viewport.

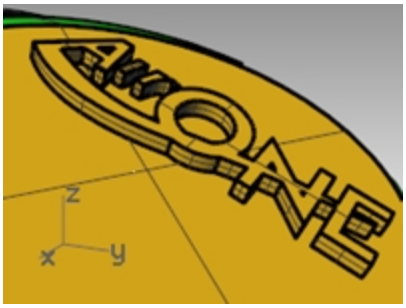
   For an accurate view of the surface and the curves relative to a custom cplane, use the **Plan** command in the **Perspective** viewport. This sets the view to a parallel projection looking straight on at the cplane.



Extrude the logo

You will extrude the logo into a polysurface.

1. Make the **Perspective** viewport active.
2. Select the remapped logo curves.
3. From the **Solid** menu, click **Extrude Planar Curve** and **Straight**.
4. In the **ExtrudeCrv** command, click to set the **BothSides=yes** option .
5. Type **1 mm** and Enter to set the extrusion height. Enter to accept and extrude.



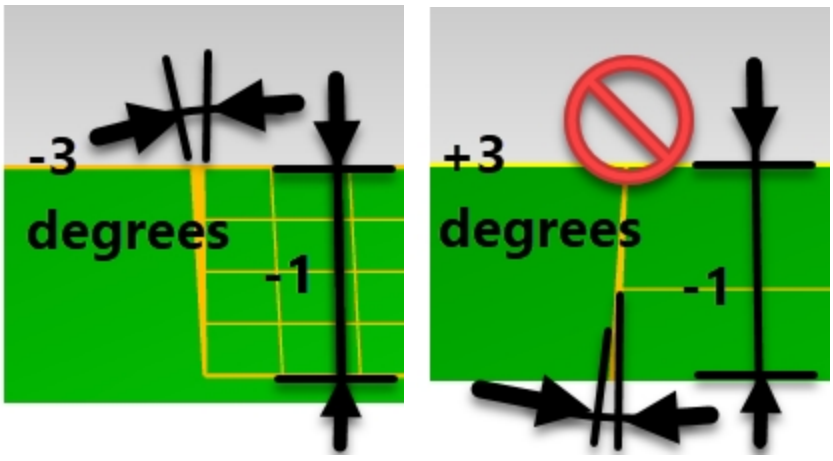
The text is now a closed polysurface. However, there is no draft angle in this extrusion that will assist in removal from a mold. Next you will **Undo** and extrude again with draft angle option.

6. From the **Edit** menu, click **Undo**.

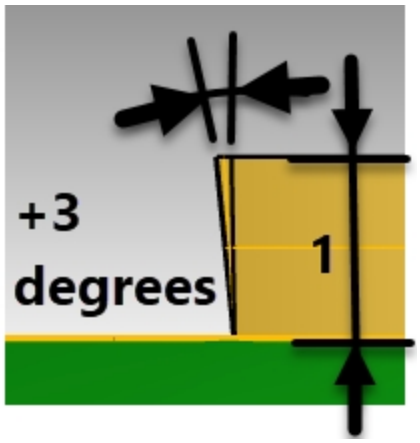
Extrude logo with tapered option

In this extrude, we will extrude the logo with the tapered option and then difference it from the top surface.

1. Make the **Perspective** viewport active.
2. From the **Solid** menu, click **Extrude Planar Curve** and **Tapered**.
3. In the **ExtrudeCrvTapered** command, click to set the **DraftAngle** to **-3** degrees.
4. Enter to complete the command.
   Setting the **DraftAngle** to posivitve**3** degrees, will produce incorrect results..



5. Type **-1 mm** and Enter to complete the extrude.



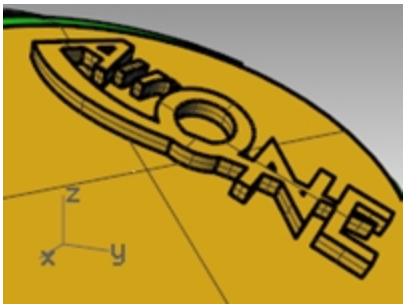
Note: Using a positive distance and a positive angle will change the extrusion height and angle to the incorrect result.

Difference the logo from the surface

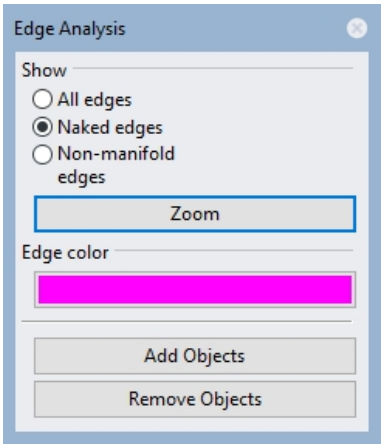
You will engrave the text into the planar surface with the **BooleanDifference** command. The **BooleanDifference** will remove the solid text from the surface and create the engraving. When the boolean is complete, check to verify that there are no openings or naked edges that were introduced as a result of the boolean.

1. Select the top surface.
2. From the **Solid** menu, **Difference**.

   Select all the extruded logo and set **DeleteInput** to **Yes**
3. Enter to complete the **Difference** command.
   Note: If the Difference fails, move the extruded text slightly above the surface and try the BooleanDifference again.



4. Pick the top surface.
5. From the **Analyze** menu, pick **Edge Tools** and **Show Edges**.
6. Select **Show Naked Edges**.



A successful Boolean Difference should result in 1 naked edge around the perimeter of the surface, and no naked edges in the interior.

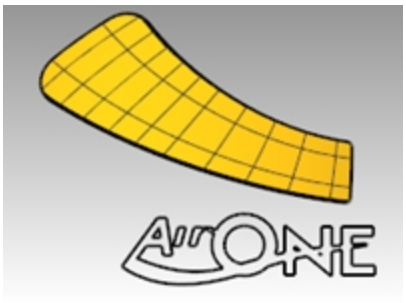
Flow the logo onto a free-form surface with history

In this part of the exercise, you will position the logo geometry by flowing it along the cutout surface. This surface is not flat so you will use a different transform tool, Flow along Surface, to move the logo and bend it along the surface. Flow along Surface morphs objects from a source surface to a target surface. It uses the UV of the surfaces to determine how it flows. It is important that the source and target surfaces have the same relative UV direction for a successful flow.

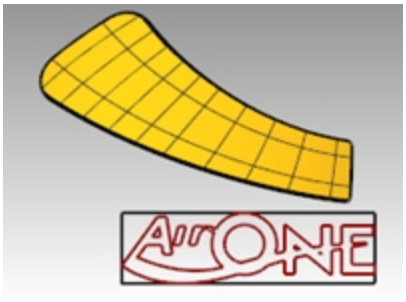
By enabling history recording before you Flow along Surface, you will be able to change the input and quickly affect the result. The logo is located on the Logo layer. You work only with the logo and the surface in this part of the exercise by turning off all the other layers.

Make the base surface

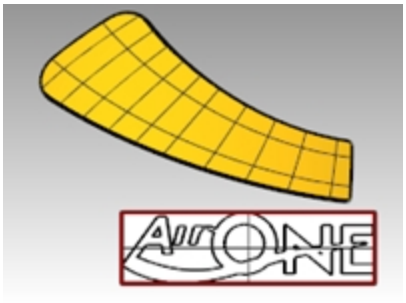
1. Start the **Layer** command and make the **Cutout** layer the current layer.
2. Turn off all the layers except **Cutout** and **Logo**.



3. Use the **BoundingBox** command *(Analyze menu: Bounding box)* to make a rectangle around the logo.



4. Use the **PlanarSrf** command *(Surface menu: Planar Curves)* to make a surface from the bounding box.



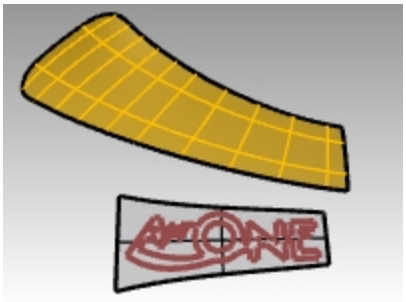
Flow the logo curves onto the cutout surface

1. On the **Status Bar**, turn on **Record History**. It will appear bold.
2. Use the **FlowAlongSrf** command *(Transform menu: Flow along Surface)* to move the logo onto the cutout surface. Pick the white surface as the base surface.
   Notice that the curve does not fit the surface.

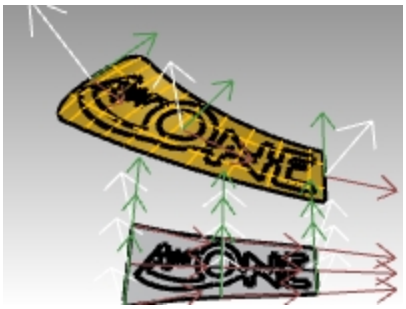
3. **Turn on the control points** on the base surface and move them to make the surface a little larger in all dimensions.
   Since history was on when flowing the curve, any adjustment to the base surface changes the way the curve fits on the cutout surface.

4. Use the **ChangeDegree** command *(Edit menu: Change Degree)* to change the base surface to **degree 3** in both the **U** and the **V directions**.
5. Adjust the control points further to fine tune the way the curve fits on the cutout surface.

6. Undo the booleans and previous FlowAlongSurf.



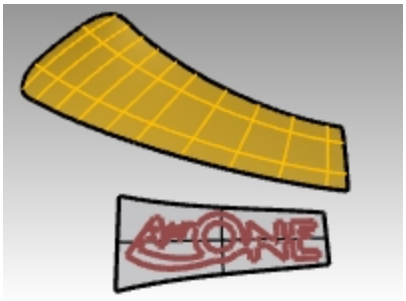
7. Use the **Dir** command to check the UV directions of the cutout surface.
8. Use the **Dir** command to adjust the UV directions of the base surface to match the direction of the cutout surface.



Raise the logo lettering and flow it onto the cutout surface

Next you will flow the solid logo onto the cutout surface.

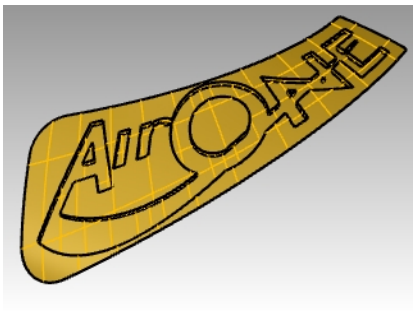
1. Use the **ExtrudeCrv** command with the **BothSides** option to make the text 3-D from the original curves. The extrusion distance should be 1 mm.



2. From the Transform menu, click **Flow along surface**.
   History is not needed this time, since all the needed adjustments were already done on the base surface.

3. Use the **BooleanUnion** command to join the logo with the cutout surface.

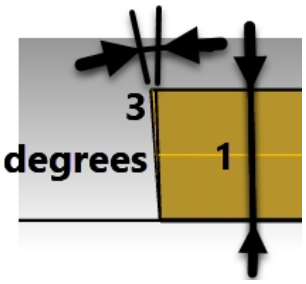
4. From the **Edit** menu, pick **Undo** .
5. Next use the **BooleanDifference** command to difference or engrave the logo into the cutout surface.



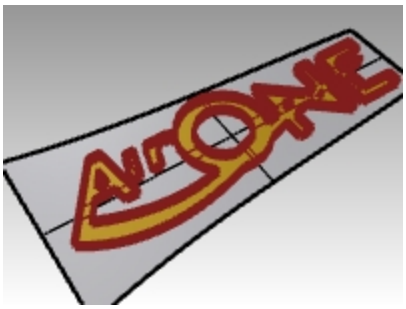
Flow the tapered logo

This time you will flow the curves instead. You will turn off the layer that contains the last FlowAlongSrf.

1. Turn on the **Cutout_02** layer and make it the current layer.
2. Turn off the **Cutout** layer.
3. From the **Solid** menu, click **Extrude Planar Curve**, then **Tapered**.
4. Set the distance to **1** mm and the Draft Angle to **3** degrees. Enter to extrude.

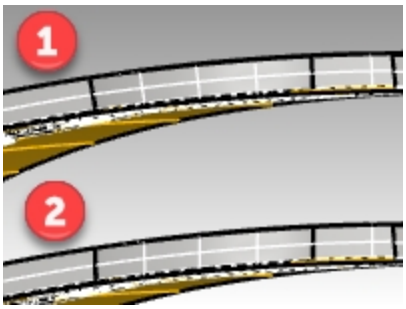


5. Group all the solid logo letters for easy selection.



6. Use the **FlowAlongSrf** command to translate the solid logo onto the cutout surface.
   Use the new base surface and click **ConstrainNormal=No**.

   The resulting flowed text should line up exactly with the previously flowed curves.
   Notice too that the embossed letters may be tapered outward in some locations. This would make it impossible to pull the surface from a mold, even though you took the trouble to taper the extruded letters originally.
   To fix this, **Undo**, and flow again, but next set **ConstrainNormal=Yes**.
7. From the **Edit** menu, pick **Undo** .
8. Use **FlowAlongSrf** command again to translate the solid logo onto the cutout surface.
   Use the new base surface and pick **ConstrainNormal=Yes**.



1) **ConstrainNormal=No**

The resulting flowed text lines up exactly with the previously flowed curves. Embossed letters may tapered outward in some locations

2) **ConstrainNormal=Yes**.

Here the flow operation maps from the flat version to the target surface without mapping to the target surface normal directions. The base vertical direction is kept.

9. Use the **BooleanUnion** command to join the tapered logo with the cutout surface.

Make a model from a 2-D drawing

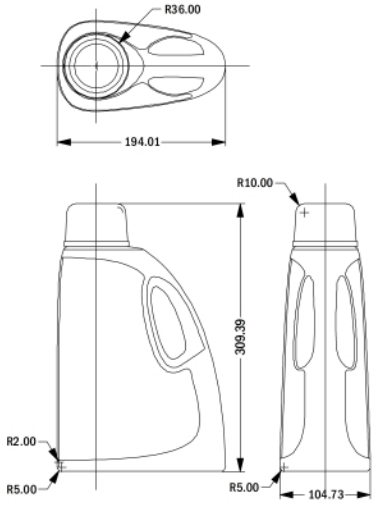
One of the more difficult modeling tasks in modeling is to interpret a set of 2-D views into a 3-D model. Very often, the drawings are precise in some areas and inexact in areas where complex surface transitions must take place in three dimensions.

It is best to consult directly with the designer to clarify difficult areas, but this is not always possible. Usually there are discrepancies between the views.

If there is no physical model available as reference, some decisions must be made along the way about the best way to interpret the control drawing. For example, you will have to consider which view to consider the most accurate for a given feature.

In the following exercise, we will explore some strategies to create a blow-molded plastic bottle from a set of 2-D drawings. In this exercise, we have a control drawing showing three views of the bottle. It is roughly dimensioned, but we need to hold to the designer's curves wherever possible.

We will only have time to finish the first stage of this model in class. We will complete the bottle surfaces, but the details will be left out. Included in the models folder is a finished bottle for your review.

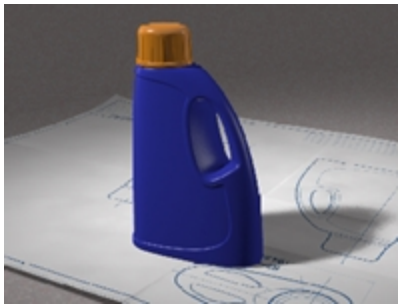


Control drawing

Exercise 11-2 Making the detergent bottle

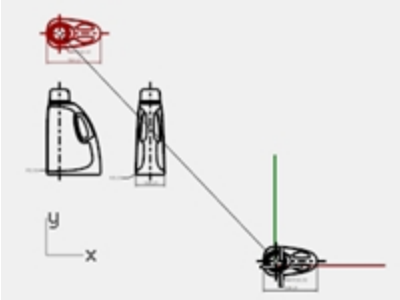
Group the parts

1. **Open** the model **Detergent Bottle.3dm**.
2. In the **Top** viewport, window select the objects that make the **Top** view (lower left) including the dimensions of the 2-D drawing.
3. Use the **Group** command to group the selected objects *(Edit menu: Groups > Group)*.
4. Repeat the previous steps to group the objects for the **Front** view (upper left) and the **Right** view (upper right). Each of the views is a separate group of objects.



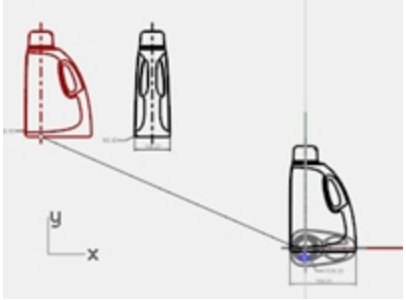
Orient the Top view

1. **Select** the **Top** view group.
2. Use the **ChangeLayer** command *(Edit menu: Layers > Change Object Layer)* to change the layer to the **2D Template Top** layer.
3. In the **Top** viewport, use the **Move** command to move the center of the circles to **0,0**.

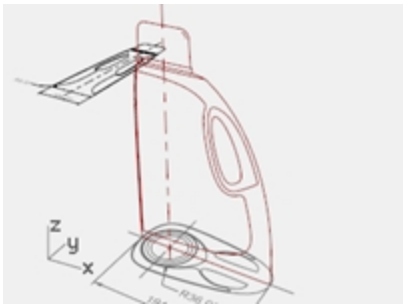


Orient the Front view

1. **Select** the **Front** view group.
2. Use the **ChangeLayer** command to change the layer to the **2D Template Front** layer.
3. In the **Top** viewport, use the **Move** command to move the intersection of the center line and the horizontal line at the bottom to **0,0**.

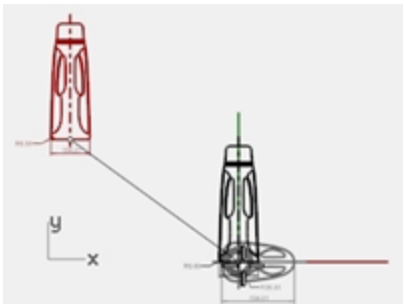


4. While the **Front** view group is still selected, start the **RemapCPlane** command *(Transform menu: Orient > Remap to CPlane)* in the **Top** viewport.
5. Click in the **Front** viewport.
   The view is oriented in 3-D space.

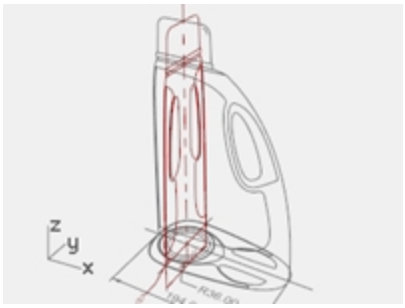


Orient the Right view

1. In the **Top** or **Perspective** viewport, select the **Right** view group.
2. Use the **ChangeLayer** command to change the layer to the **3D Template Right** layer.
3. In the **Top** viewport, use the **Move** command to move the intersection of the center line and the horizontal line at the bottom to **0,0**.



4. Use **RemapCPlane** to map the **Right** view curves to the **Right** construction plane.
   The view is oriented in 3-D space.



   Frequently 2-D curves for design control drawings will not be as carefully constructed as you like for making accurate geometry. Before building 3-D geometry from the 2-D curves, check the curves and correct any errors that can be found.

Create the 3-D curves

The inset part of the bottle will be cut into the surface later. For the moment, we just need to build the outer surfaces. The fillets at the top and bottom indicated in the curves can be left out of the initial surface building and added in as a separate operation. We will need to extend or redraw the edge curves to bypass the fillets and meet at hard corners before making the surfaces.

Several surfacing tools could be used to build the initial surfaces: A **Two-Rail Sweep** or a **Surface from Network of Curves** are the obvious choices.

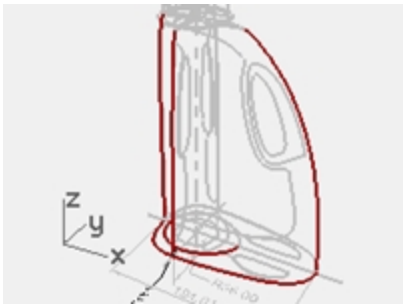
Network surfaces do not pay any attention to the curve structure, only the shape. All curves are refit and the resulting surface has its own point structure.

Other commands including the Sweep tools, lofting and edge surfaces do pay attention to the curve structure in at least one direction. In these cases, it often pays to use matched curves as cross sections. The choice of surfacing tools may well determine the way in which the actual input curves are created.

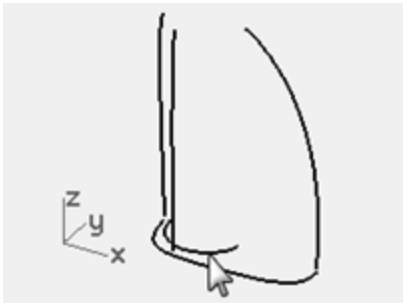
1. **Select** the groups you made in the previous step, use the **Ungroup** command *(Edit menu: Groups > Ungroup)* to ungroup them.
2. **Select** the curves from each 2-D template view that define the outer surface and **Copy** them to the **3D Curves** layer.
   Since the bottle is symmetrical on both sides of the X-axis, you will only need to copy the curves on one side.

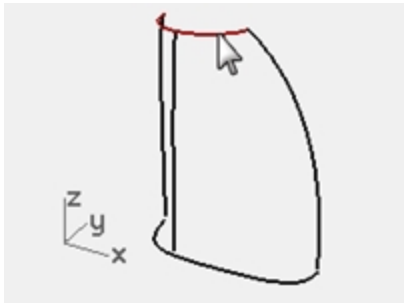
They will be mirrored later.

3. Use the **OneLayerOn** command *(Edit menu: Layers > One Layer On)* to set the **3D Curves** layer.

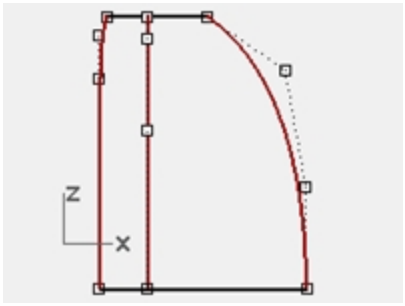


4. **Move** the curve defining the top surface of the bottle to the same height as the top of the vertical curves.
   Use **SetPt** or **Move** with the **Vertical** option in the **Perspective** viewport.

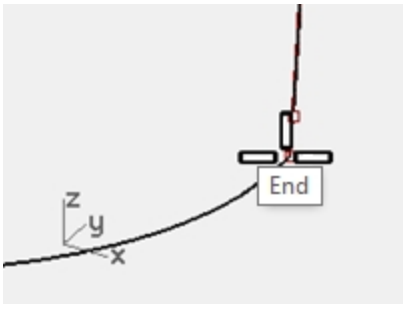




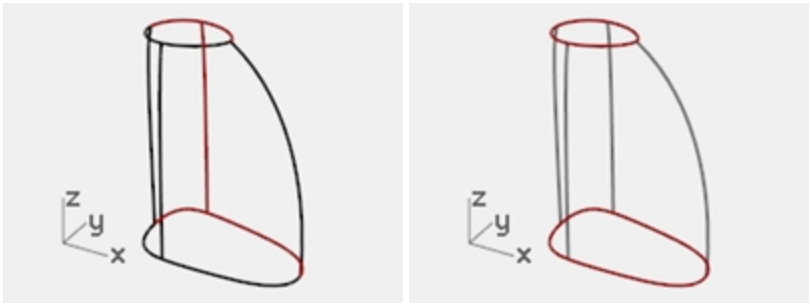
5. The vertical curves now can be extended past the fillet curves so that they meet the top and bottom curves exactly on the endpoints of these curves.
   One way is to extend the vertical curves using **Extend** with **Type=Smooth**.
   Snap to the **End** or **Quad** points of the top curve and the base curve at the bottom.
   Extending the curves in this way will add complexity to the curves. If it is important to keep the curves simple and well matched, it may be better instead to adjust the points on the existing curves to extend them.



6. **Undo** the **Extend** operation, and instead point edit the curves directly.
   You can make a duplicate set of curves and edit one of each leaving the original in place as a template.
7. **Mirror** the base, top and side curve visible from the **Right** view to the other side.
   The result should be a set of eight curves that define the surface.
   Most of these curves are essentially the original curves from the 2-D drawings but rearranged in 3-D.



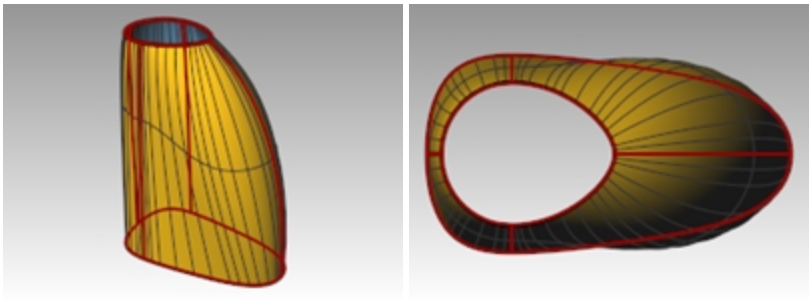
8. **Join** the base curves and the top curves into a closed loop.
   The curves are set up for a surface from a curve network or a two-rail sweep.



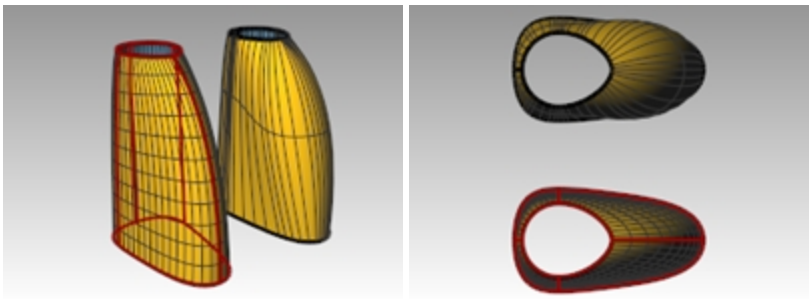
Make a surface for the bottle with a sweep

From the drawing these are the only curves we have available to define the shape, so we will use these curves directly to create the surface.

1. Make the **Surfaces** layer current.
2. **Window select** the curves, and try **Sweep2** to make a surface then **Shade** the viewport.
   Notice the shape gets severely out of control at the rounded side of the bottle.



3. **Move** this surface to the side for the moment.
   While it is possible to rearrange or add curves to make the **Sweep2** work better, it is worth checking how a surface from a **Curve Network** will work with the same set of curves.
4. **Select** all of the curves, again, then use the **NetworkSrf** command to create the surface.



The NetworkSrf command handles this set of curves much more gracefully.

On your own

- Make the inset surface and the handle.
- Fillet the edges where indicated in the 2-D drawing.
- Model the threads and cap

A finished bottle, **Finished Detergent bottle.3dm**, is included in the Model folder for your review.

Chapter 12 - Surface analysis

Rhino has several tools to help you visually evaluate surface quality. In this exercise, we will use curve and surface analysis tools to help build clean, simple surfaces with good continuity.

Some models require more attention to continuity, primarily, because it will show when manufactured. For example, a blow-molded bottle will not show slight inconsistencies in the surface, but a car panel will.

The file, **Surface Analysis.3dm**, has a set of curves you will recognize from the previous exercise. Our task in the current exercise is to make curves that can create cleaner, simpler surfaces with good continuity. We will make use of the **CurvatureGraph**, **Zebra**, and **CurvatureAnalysis** to make sure we set things up for the best results. Lastly, we will compare the surface analysis results in this file with the surface analysis results in the previous exercise.

Exercise 12-1 Surface analysis

Evaluate the curves

First, let us look at the curvature graphs of the top and the bottom curves. These curves have a nice enough shape, but there is room for improvement from a surfacing continuity perspective.

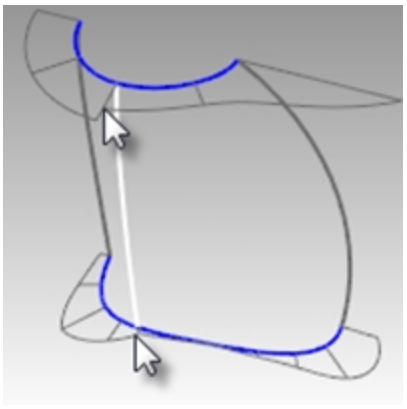
Open the model

1. Open the file **Surface Analysis.3dm**.
2. **Select** the top and bottom curves.
3. Start the **CurvatureGraph** command *(Analyze menu: Curve > Curvature Graph On)* and set the **Display scale** to a value of **120**.

The graph tells us that both curves are tangent continuous but have curvature discontinuities in a couple of locations.

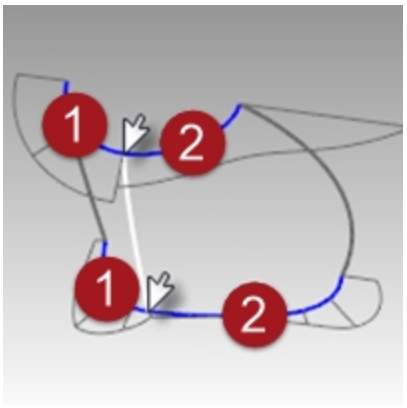
Assuming we want the surfaces we build from these curves to have curvature continuity throughout, we will be better off modifying these curves before creating the surfaces.

If we decide ahead of time what the surface arrangement will be, that will help us understand how to draw clean new curves.



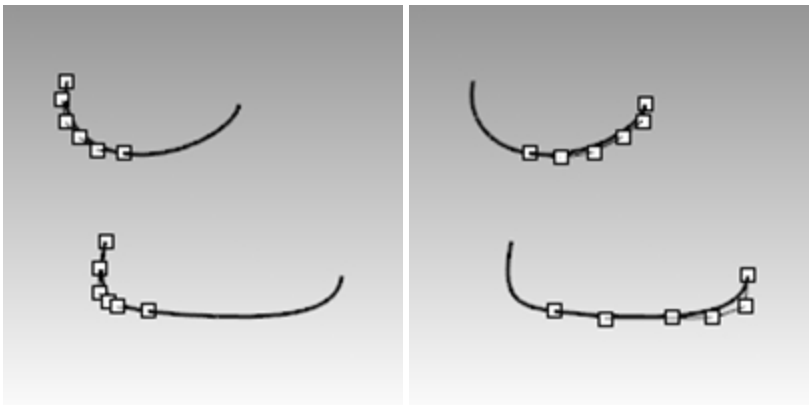
Building a surface that has consistent curvature as a single surface is good modeling practice. Looking at the bottom curve, we can see two areas that are good candidates for creating the surfaces, so we will start there. There is an area of high curvature at the front (1) and a relatively flat stretch in the middle with a rapidly increasing curvature at the handle side (2). The top curve is smoother overall, but has similar corresponding regions of curvature.

By examining the current curves, we can identify two curves to build at the top and bottom. The white vertical curve intersects the top and bottom curves at the curvature discontinuity of the top curve, which is a modified circle, and on an abrupt change in curvature on the lower curve. This intersection is where we will start and end our modified curves.

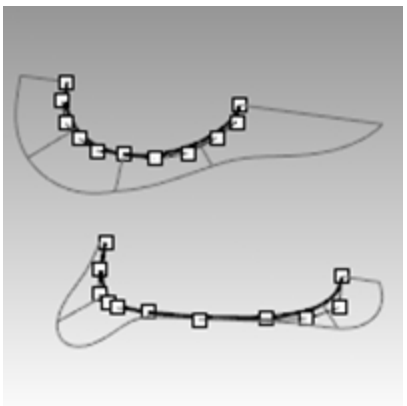


Build the modified curves

1. Using the current curves as a reference, draw four new curves of degree five with six points.
   The goal is to redraw the top and bottom curves in two parts each.
   Keep in mind what you know about continuity, CurvatureGraph, tangent directions, and EndBulge.
   Try to keep the control point locations even and progressive.



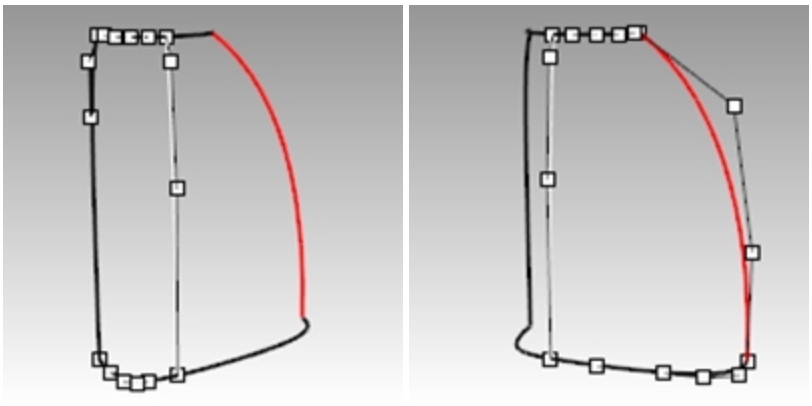
2. Analyze your curves with the **CurvatureGraph** command.
   Try to get the graph clean with minimal, abrupt changes, while at the same time match the original curve shapes as closely as possible.
   They cannot be exactly the same as the originals if they are to have better continuity, but it should be possible to get close.



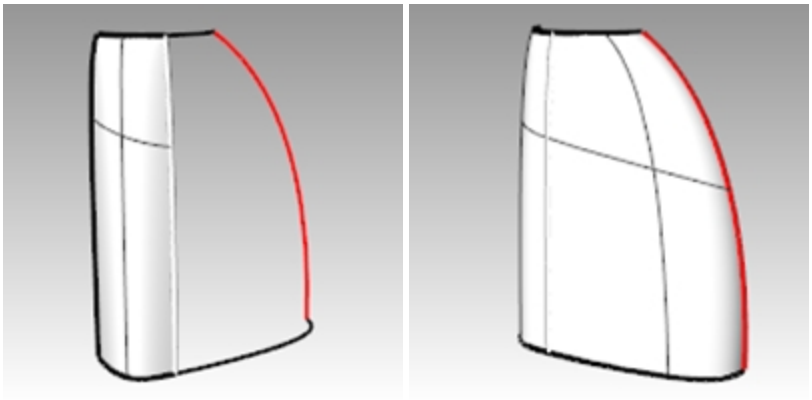
Make the surfaces for the bottle from edge curves

There are four, single-span, curves that define each area for the surfaces. In this part of the exercise we will use the **EdgeSrf** command *(Surface menu: Edge curves)* to create the surfaces. This command is one that uses the input curve structure to create the surface. It works best if the curves on opposite sides of the rectangle match each other. The resulting surface will be simpler.

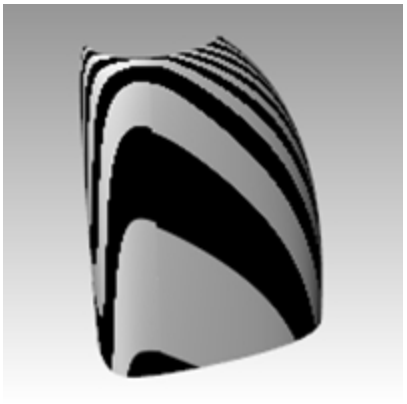
Since we have taken care to meet this criterion, all of the vertical curves are degree 3 with four points, and the curves we just made are degree five with six points, the resulting surfaces will share this structure.



1. **Select** four curves that define one of the surfaces.
2. Start the **EdgeSrf** command *(Surface menu: Edge curves)*.
3. Repeat steps 1 and 2 for the other surface.

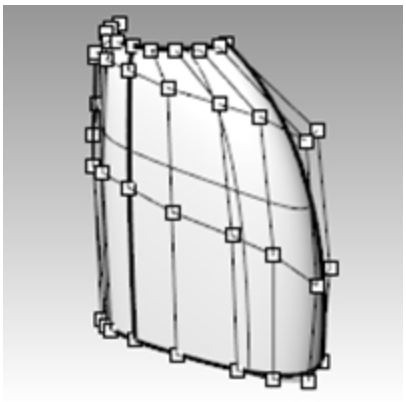


4. Check the surfaces with the **Zebra** command.
   The zebra stripes have a nice even flow but the surfaces are clearly not tangent at the vertical edge.

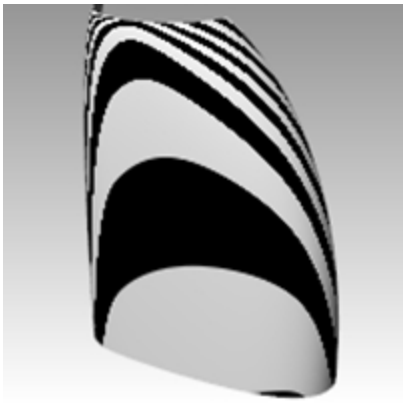


Match the surfaces for the bottle with MatchSrf

1. Use the **MatchSrf** command *(Surface menu: Surface Edit Tools > Match)* to match the surfaces for **Curvature**. Try matching in both directions, and with or without **Average surfaces** set.
   In this case, the results are good no matter how the match is made, but it is worth looking at the control points on the surfaces in each case.
   Matching the larger surface to the smaller one, without **Average**, results in a more erratic control point arrangement on the larger surface than any other combination, particularly the second row from the top. Other considerations being equal, the best choice is the surface with the most regular, even control point arrangement.



2. Check the surfaces with the **Zebra** command.
   The zebra stripes have a nice even flow with no discontinuity at the common edge.

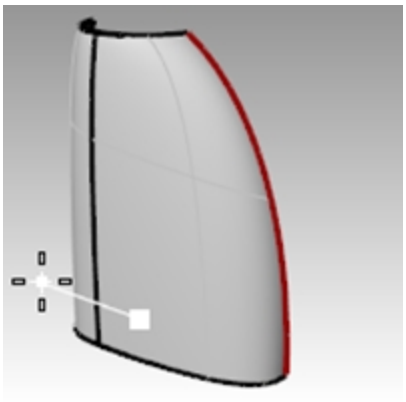


Match the surfaces for the bottle with symmetry

In this section, we will make the other half of the bottle using the **Symmetry** command with **Record History** turned on. Symmetry mirrors curves and surfaces, makes the mirrored half tangent to the original, and with History recording active, when the original object is edited, the mirrored half updates to match the original.

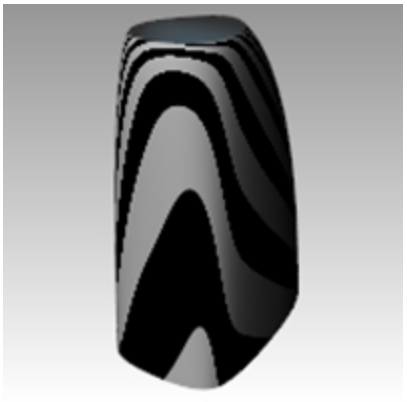
1. **Select** the larger surface.
2. Use the **Symmetry** command *(Transform menu: Symmetry)* to mirror the surface across the x-axis.
3. Turn **Record History** on.
4. For the **Select curve end or surface edge**, select the edge of the surface (1).

5. For the **Start of symmetry plane**, type **0** and press Enter.
6. For the **End of symmetry plane**, use **Ortho** to pick a point along the x-axis.



7. Repeat this process for the other surface.
   If you edit either original surface, the mirrored part will update to match.

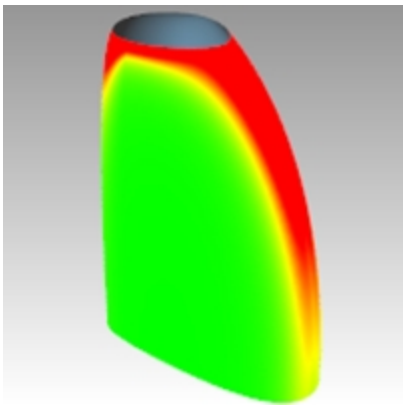
8. Check the surfaces with the **Zebra** command.
   The zebra stripes have a nice even flow with no discontinuity at the common edge.



Analyze the matched surfaces

At this point, we will use the **Curvature Analysis** tool to evaluate the matched surfaces. This can be useful in locating areas of extreme curvature, but may force the display to ignore more subtle changes. In any case, the display on each of these simple surfaces should be very smooth and clean.

1. **Hide** all curves to get a good view of the transitions between surfaces.
2. **Select** all of the surfaces and turn on **Curvature Analysis** display *(Analyze menu: Surface > Curvature Analysis)*. Set the style to **Gaussian**, and click **Auto Range**. Make sure you have a fine analysis mesh for a good visual evaluation.
3. Click back and forth between **Auto range** and **Max range**.
   Auto Range attempts to find a range of color that will ignore extremes in curvature, while **Max Range** will map the maximum curvature to red and the minimum to blue.
   The numbers are for **Curvature**, which is, 1/radius.

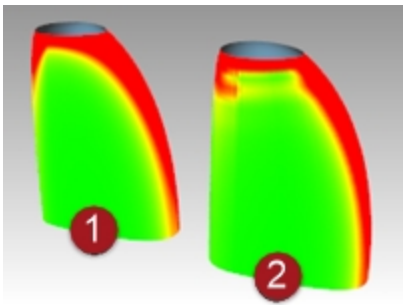


The goal when matching is to maintain as even and gradual a curvature display as possible, while meeting the continuity requirements.
Notice the edges that have been matched appear to have a smooth color transition.

Analyze and compare different surfacing techniques

Next, we will make another surface for comparison.

1. **Copy** the curves to one side.
2. **Mirror** and **Join** the top and base curves on the x-axis.
3. **Mirror** the vertical side curve on x-axis to make a set of curves suitable for a surface form a curve network.
4. Use **NetworkSrf** to make a surface from these curves.
5. Select the new surface and **Add** it to the **Curvature Analysis** display.
   The denser network surface (2) has a less clean appearance in this display. The simple surfaces (1) still look cleaner at this point.
   Since the color change is mapped across the entire range shown, it is important to remember that the Auto Range setting indicates a very narrow range of curvature and that the actual differences may be small even though the color change is great.



Chapter 13 - Sculpting

Designers can build a relatively undefined surface and then use a variety of transform and analysis tools to sculpt a surface in 3-D space in an intuitive and direct manner—design "as you go."

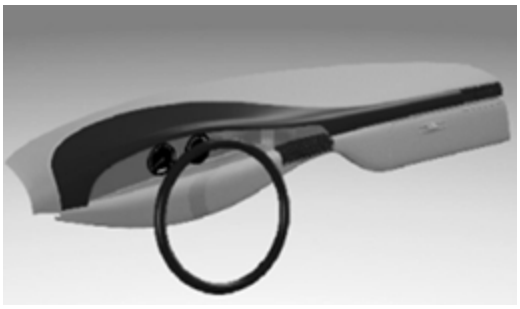
Place curves approximately. The curves should be edited copies of a single original if possible.

This ensures that they will be compatible when lofted, and create the simplest, most easily edited surface.

Start with the big changes and then make the detailed changes.

Use the **IncrementalSave** command to make copies of the model as you progress.

In the following exercise, we have created four curves for you to use. They describe a very simple dashboard from which to start your design. An added steering wheel, on a locked layer, is to help you get a sense of scale and positioning of any elements you might wish to add.



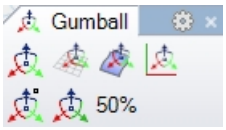
Tools to help in control point editing

Move Control Points by dragging, using the **Move**, **Rotate**, or **Scale** commands or other transform commands.

Dragging is the most fluid and interactive way to edit the shape. However, there may be situations where the construction plane constraint for point movement is less than optimal. Here are some tools that can help with control point editing:

Gumball

Gumball works very well with control points. It is convenient to be able to switch easily among the various gumball orientation modes. Opening the Gumball toolbar will help.



When **Gumball orientation** is set to **Align to Object**, the blue axis orients itself to the surface normal for a selected control point. By dragging the blue arrow, the control point moves in the local surface normal direction. The red axis aligns to the surface U direction. Make fine adjustments by setting the Gumball drag strength to something less than 100%.

DragMode

DragMode lets you override the current construction plane constraints in several ways.

DragMode command-line options

World

This constrains dragging as if the construction plane were always World Top. This is rarely used.

CPlane

The default constraint in Rhino. Dragging takes place parallel to the active construction plane.

View

Dragging takes place parallel to the current viewport. This can be useful in some oblique views.

UVN

Dragged surface control points with **Ortho** on (Shift key) are constrained to the surface u- and v-directions and with the Ctrl key are constrained to the surface normal direction. Curve points are constrained to the curve tangent and with Ctrl to the curve normal direction. For this exercise, this is most useful drag mode.

ControlPolygon

This mode constrains dragged curve and surface control points to the control polygon. In the case of multiple selected points, each moves along its own control polygon. This is an excellent tool for keeping points well organized in rows and columns.

As with the Gumball, having the **DragMode** toolbar open makes it easy to switch among modes while point editing. Notice the cursor changes to reflect the current drag mode. In general, you will probably find it easier to switch the Gumball off when dragging points using the special drag mode options.

MoveUVN

This tool opens a dialog box that allows you to move control points in increments according to a user set scale. The point movement can be along the U, V, and N (Normal) directions. In addition, there are smoothing tools in this dialog. These are very useful for smoothing out bunched or irregular control points to achieve a more regular grid.

Nudge

Use the arrow keys with Alt, Alt + Shift and Alt + Ctrl to move points in small increments. Note the settings in the *(Options > Modeling aids > Nudge page)* allow you to set the Nudge constraints similarly to some of the DragMode settings mentioned above.

Tip: You can use the knowledge gained in the User Interface portion of this course to create macros that make it easy to swap among the nudge modes.

SetPt

This allows you to true up points or rows of points in one, two, or all three dimensions.

Use any or all of the above tools to manipulate the surface points individually and in groups. Keep in mind the point selection tools: **SelU**, **SelV**, and others in the **Select Points** toolbar.

You will see that with a relatively sparse set of control points, as we have on this starting surface, the edits you make tend to be large ones that affect the shape broadly. It is likely that you run out of control quickly.

In this case, you will find that you need have more localized control to add smaller details. You can add control by increasing the density of the control points. There are two similar tools for this:

InsertKnot

Inserts one or more knots, and rows of control points. Surface points are rearranged so that the shape of the surface does not change. In other words, except in special cases, the new points are not added at the same location as the new knot.

Some considerations when inserting knots

Insert as few knots as you can to get the control you need. Add more later if needed. Keep the surface as simple as possible while getting the control you need.

Where possible, insert knots so that they are as evenly spaced as possible. Try to place them halfway between existing knots.

InsertControlPoint

Allows you to place the row of points where you want them to be, however the mechanism for adding these points does nothing to ensure that the shape of the surface stays the same. Generally, the shape will change.

Both **InsertKnot** and **InsertControlPoint** have advantages, but when you are making heavily curved and sculpted surfaces, InsertKnot may be the safer choice since it does not change the shape of the surface.

InsertControlPoint command-line options

Automatic

Adds knots halfway between existing knots to maintain as uniform a structure as possible. It increases the knot density of a curve or surface, and thus the control point density as well, while maintaining an even knot distribution, which makes point editing more predictable than with unevenly spaced knots. Be careful with this option, as it adds a knot between each pair of existing knots, so a surface can become very dense in structure very quickly.

MidPoints

Places markers halfway between the existing knots/knot lines that can act as guides for inserting knots midway between existing knots. Use the Point object snap to place new knots at these markers.

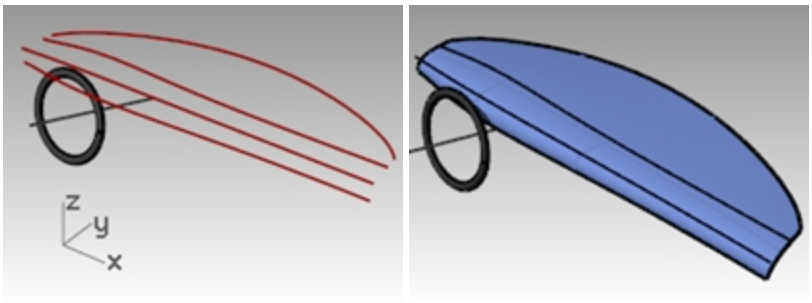
Note: It is easy to get to a point where there are too many control points and you simply lose control of the shape. Therefore, it is a good idea, to use IncrementalSave before adding complexity to the object. This will allow you to go back to a simpler model without starting over if things get out of hand.

Exercise 13-1 Dashboard

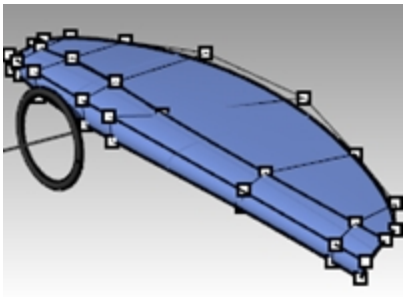
In this exercise, we will build a car dashboard using control point editing.

Loft the construction curves

1. **Open** the model **Dash.3dm**.
2. **Loft** the four curves together with the Loose option from the drop-down list.
   Using the **Loose** style creates the simplest possible geometry and is essential to creating a surface with this technique. The surface will not touch the interior curves of the loft with this option, but it should be very smooth and clean looking.

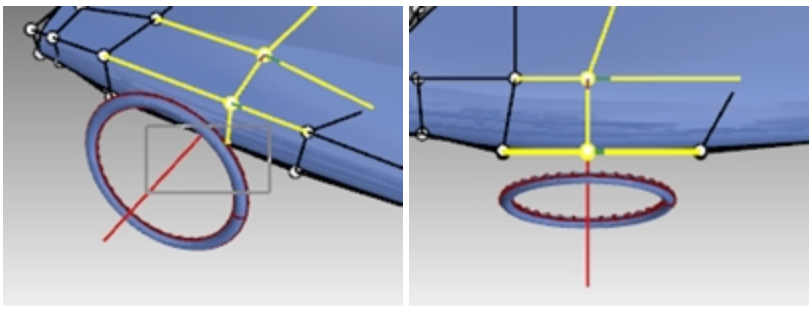


3. Turn on the control points.
   If you also turn on points for the input curves, you will see that the point structure of the surface exactly matches that of the four curves.
4. Turn off the **Curves** layer.
   At this point, you can sculpt the surface directly by transforming the surface control points.
   As mentioned previously, Rhino has several powerful tools to help in control point editing.



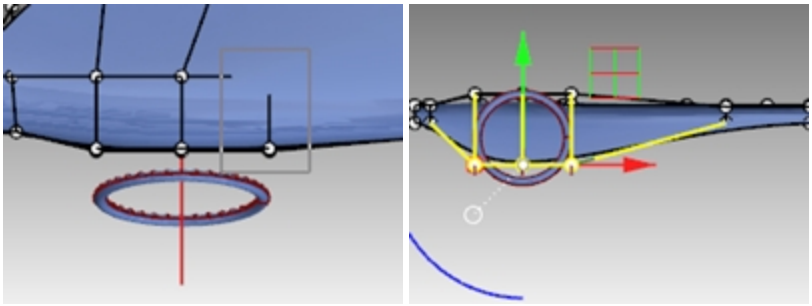
Edit and sculpt the surface

1. **Turn on the points** for the surface, with Gumball off for the moment.
2. **Window select** the three control points that are just to the right of the steering column center line.
3. Use the **SetPt** command *(Transform menu: Set XYZ Coordinates)* to align the groups of points in the x direction in **Top** or **Front** snapping to the center line.

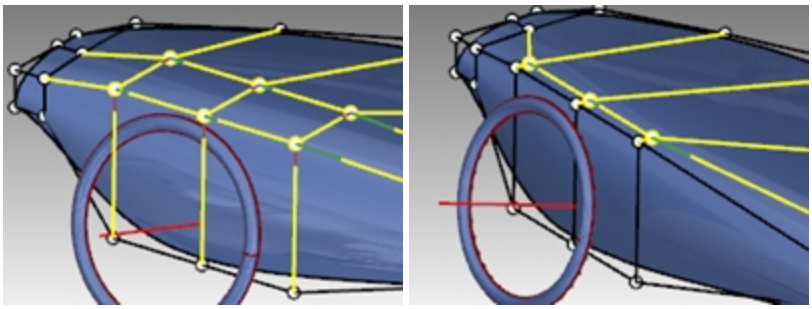


Notice that when the points are selected, there is a red and a green line extending from the selected points- this indicates the positive u- and v-directions.

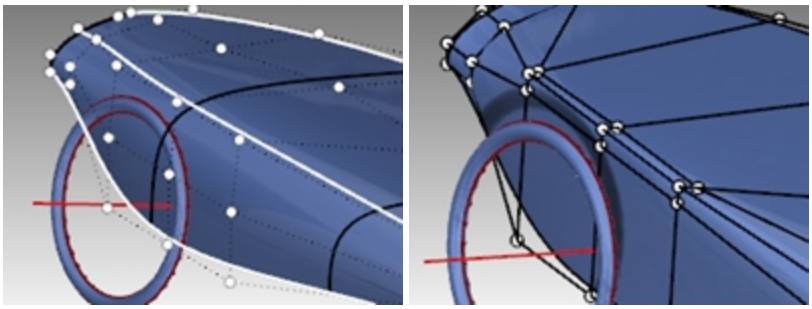
4. Use this information with the point selection tools **NextU**, **NextV**, **PrevU**, or **PrevV** to shift the selection to the left or right to the next row of points. **SetPt** these points to the corresponding edge, left and right, of the steering wheel.
5. Use **Gumball** to drag the lowest three points near the steering wheel down to accent the shape.
   The shape may not be symmetrical compared to the steering wheel.



6. Select the points nearest the top edge of the steering wheel and set them all to the same **Z** using **SetPt**.
7. Now, let us add some definition to the dash in the area of the steering wheel.
   Set the **DragMode** to **ControlPolygon**, and drag three of the points up very close to their neighbors.
   The shape of the surface changes only slightly. It is still very soft, and there are no more points available to continue to edit there.



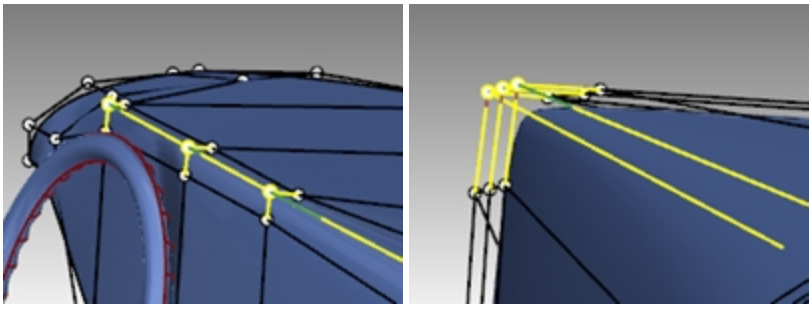
8. Use the **InsertKnot** command *(Edit menu: Control Points > Insert Knot)* to add a row of points in the v-direction.
   The surface currently has no interior knots so use the Automatic option once to add a single knot.
   Now, with the extra points you can continue to slide points along the control polygon to sharpen up the surface in this area.



9. Use the **IncrementalSave** command *(File menu: Incremental Save)* before going on to the next step.

Another way to tweak the shape

1. Set the point weight of some of the points on the surface.
2. For example, select the three middle points that are at the apex of the now sharply curved area we have been editing.
3. Start the **Weight** command *(Edit menu: Control Points > Edit Weight)*. Increase the weight of the points to 2. Higher weight points, relative to neighboring points, tend to pull the surface toward them. Lower weight points cause the surface to fall away from the control points.

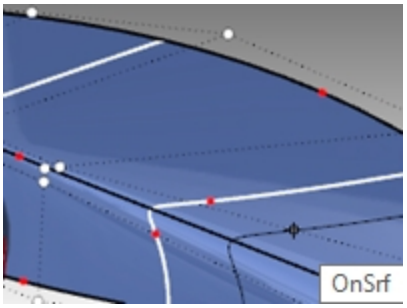


Add knots

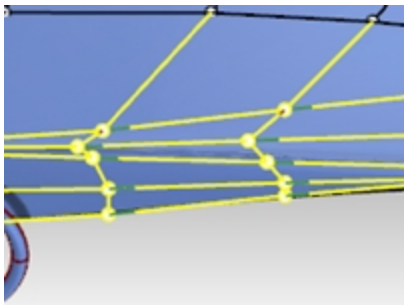
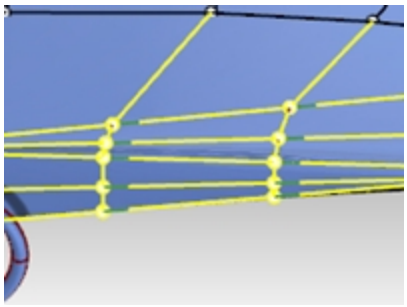
Try inserting knots in either direction, using the **Midpoints=Yes** setting to be able to snap to span midpoints. Insert some knows with **Midpoint=No** to insert knots close to existing ones to allow more local control.

Add and manipulate knots

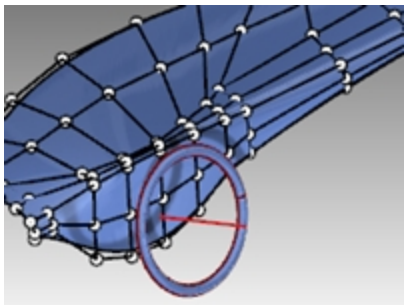
1. Use the **IncrementalSave** command *(File menu: Incremental Save)* before going on to the next step.
2. To insert some knots in the u-direction, use **InsertKnot**, **Midpoints=Yes**.
3. Snap to midpoints to keep knot spacing even.
   Notice how the control point arrangement changes with each knot inserted. You may want to true up some rows of points in X using **SetPt** to keep the rows and columns organized.
4. Explore different shapes and design ideas using all of the tools mentioned above.



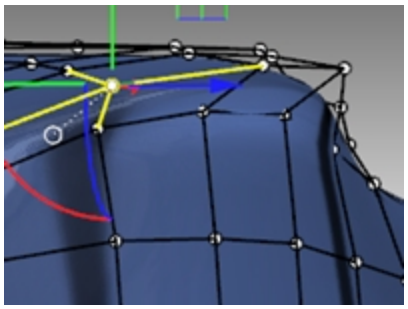
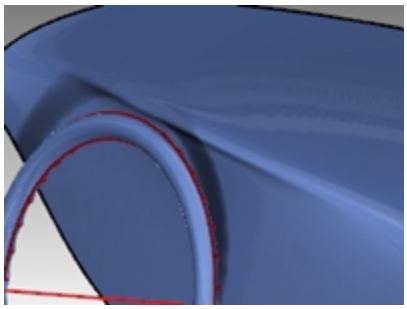
5. Select some points that are out of alignment. Start the **MoveUVN** command *(Transform menu: MoveUVN)*.
6. Use the **Smooth** sliders to smooth some of the points that are out of alignment.

7. Move some points to the right of the steering wheel closer to it to make the shape more symmetrical about the steering wheel.
   Where possible, try to keep the control point arrangement even and progressive. As you might expect, there tend to be more control points in areas that have been edited and shaped the most.



8. Add some more knots in the area of the top of the steering wheel.
   If you crowd a few knots in this area, you can pull out a localized feature that fades smoothly into the surrounding surface.

Add details

When you are satisfied with the overall shape of the surface, you can add details to make a more finished object.

The surface can be offset and trimmed as in the first illustration.

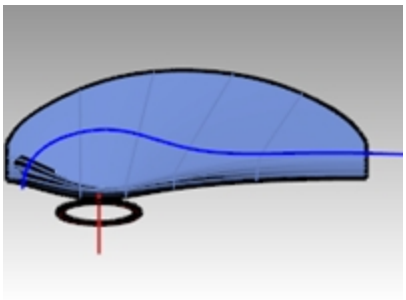
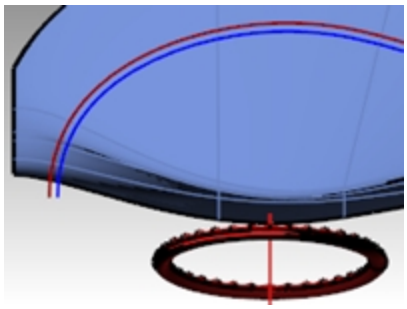
Best results are obtained when the surface has at least degree-3 in both directions. Check this with Object Properties.

Note: Offsetting surfaces generally results in a surface of one step lower in internal continuity. Surfaces that are only G1 internally may result in surfaces that have G0 continuity; that is, they may have a kink in them. Although Rhino allows these surfaces, this can lead to problems downstream.

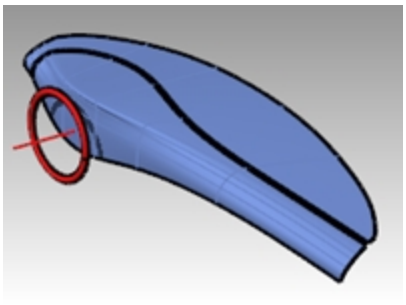
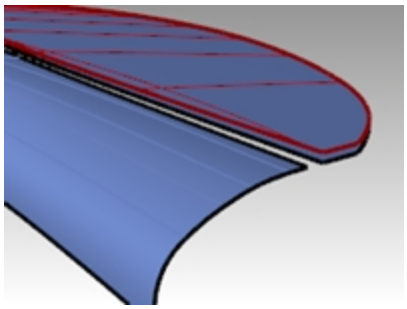
For this reason, if you intend to offset surfaces, it is best where possible to create the initial surface from degree 3 or higher curves. These surfaces have at least G2 continuity so that offsetting them will result in at least G1 continuous surfaces. Changing the degree of a surface that has been created from degree-2 curves to at least degree 3 in both directions is not sufficient to ensure a G2 surface. Simply changing the degree after the fact does not improve internal continuity.

Make the offset surface

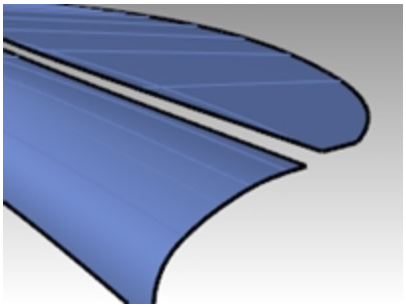
1. Change to the **Cutting Curves** layer.
2. Draw a curve that represents where you want to split the surface.
3. Use the **Offset** command *(Curve menu: Offset Curve)* to make a duplicate of the curve offset by one-half (0.50) inch.

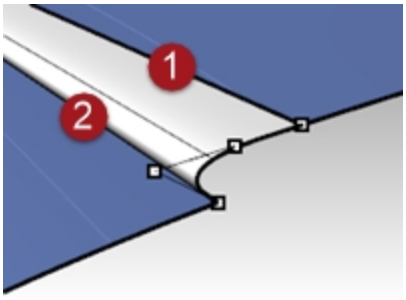
4. Use the **Trim** command *(Edit menu: Trim)* to trim the surface between the curves.
5. Use the **OffsetSrf** command *(Surface menu: Offset Surface)* to offset the back surface by one-fourth (0.25) inch.

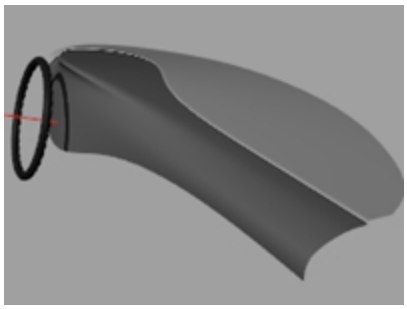



6. **Delete** the original surface.

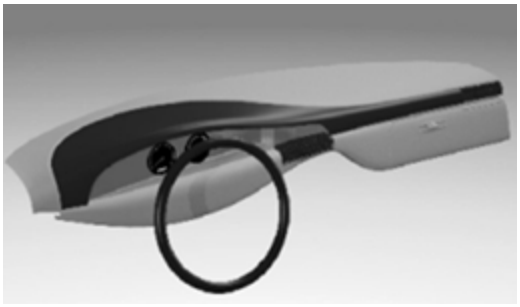


7. Use the **BlendSrf** command *(Surface menu: Blend Surface)* to blend between the two surfaces.
   Set **Edge 1** to **Curvature (G2)** and **Edge 2** to **Position (G0)**.
   One of the things we are trying to show here is a quick way to make a "tucked" upholstery type transition.
   Adjust the **BlendSrf** sliders so the cross-section looks like the example on the left.





8. Add details if time allows.



Chapter 14 - Deformation tools

Deformation tools allow you to deform meshes, lines, surfaces, polysurfaces, and solids without worrying about the integrity of the object.

Start the deformation commands from the **Transform** menu, or from the **Deformation Tools** toolbar:

Deforming objects

With CageEdit the deformation of captive objects will take place throughout 3-D space. This setting is important if some of the captive objects are outside of the cage.

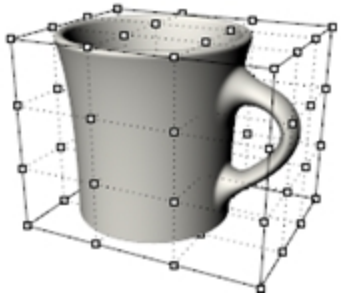
CageEdit

Exercise 14-1 Using cage editing to deform an object

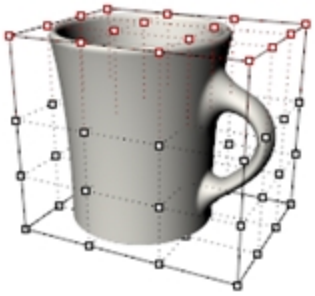
Deform an object with CageEdit

1. Open the model **CageEdit_Mug.3dm**.
2. Open the **Cage** toolbar.

3. Start the **CageEdit** command *(Transform menu: Cage Editing > Cage Edit or Cage toolbar)* and select the mug as the captive object.
4. For the control object, choose **BoundingBox**, and then **World**.
5. Specify four cage points and degree 3 in each direction.
   The degree can be set up to degree 9. The point count must be greater than the degree value in each direction.
6. For the **Region to edit**, choose **Global**.



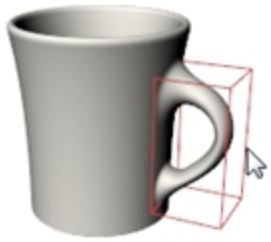
7. Move the control points on the top of the cage vertically to deform the mug.
   Cage points can be moved, dragged, scaled, sheared, rotated, bent, etc.



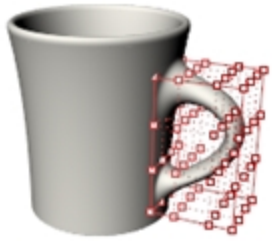
Limit the region to edit

To deform the handle only, a cage object is needed around the just the handle.

1. Use **ReleaseFromCage** command *(located in the Cage toolbar)* to select the cage object on the mug.
2. Enter to **Delete** the cage.
3. Start the **Cage** command and create a freestanding cage object.
4. **Move** the cage into place and **Scale** it before attaching it to the mug
5. Start the **CageEdit** command.
6. For the **Select captive object**, select the mug.
7. For the **Select control object**, select the pre-positioned cage object.
8. For the **Region to edit**, choose **Local** and set a **Falloff distance** to **5**.
   This tells Rhino that the distortion of the cage will affect only the region inside the cage, plus a smooth falloff effective over 5 units moving out from the cage itself.



9. **Move** the two right-most vertical rows of cage points slightly to the right in the **Front** viewport to increase the size of the handle.



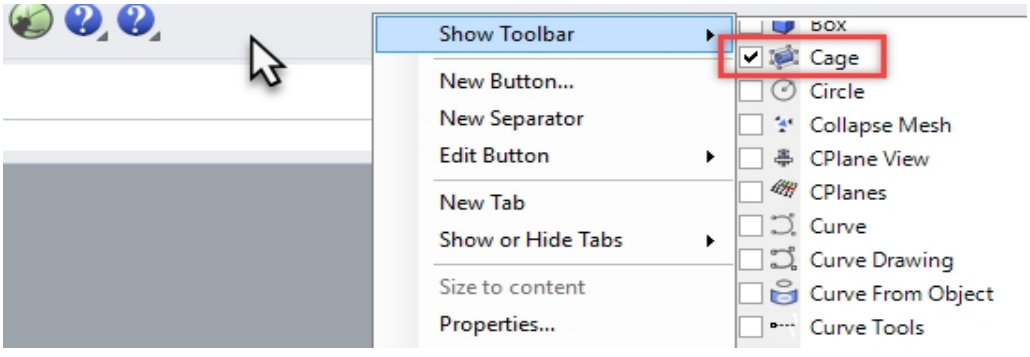
Exercise 14-2 CageEdit the salad fork

While it is possible to use any curve or surface as a control object in **CageEdit**, there are many cases where the most intuitive solution is to use a curve or surface that is already part of the object.

CageEdit with a surface from the object

In the**CageEdit** command you will select a surface from the captive object to use as control object.

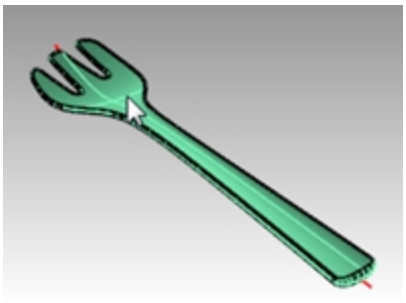
1. Open the model **Salad_Cage.3dm**.
2. **Right-click** the end of the **Standard** toolbar.
   On the menu, click **Show Toolbar** and check **Cage**.



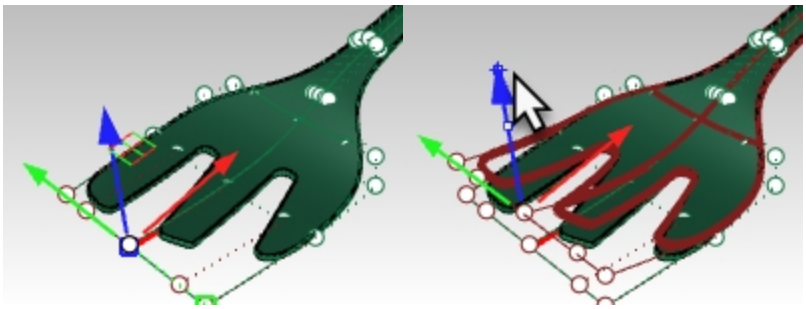
   The Cage toolbar will appear. 

3. Select the salad fork and start **CageEdit** command.

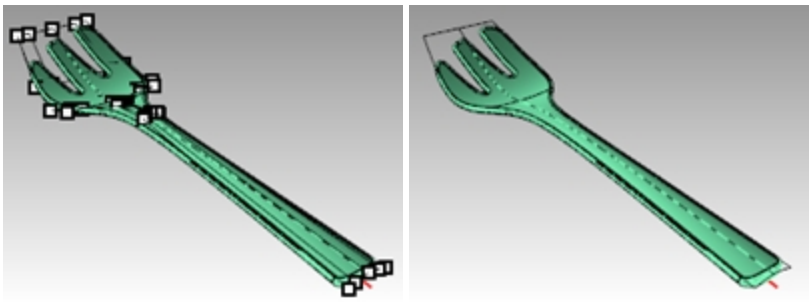
4. For the **Control object**, click on the top surface of the fork.



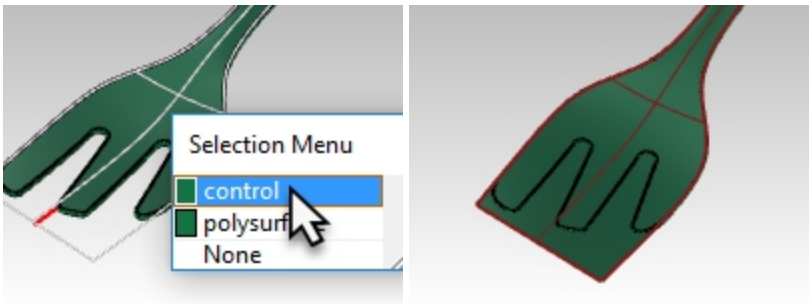
5. At the **Region to edit** prompt, pick **Global**.
   A copy of the surface is extracted from the object and turned into a control object.
6. When the control points are on, use the Gumball to move the five points nearest the end vertically.



   This gives the fork a little more bend to the front of the fork. This typically results in considerable delay and a useless distorted version of the fork.
   The reason for this is that the extracted surface does not extend to the full width and length of the fork- the rounded, blended edges fall outside of the control object and therefore it becomes highly distorted as a result.
7. **Undo**.

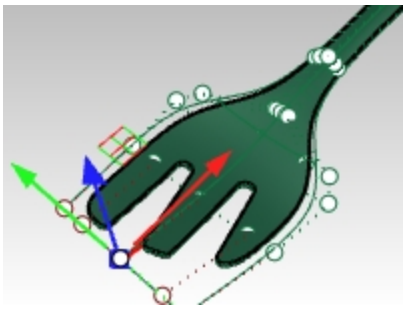
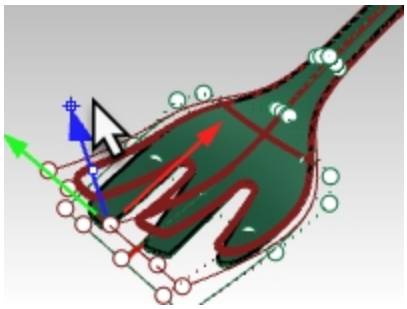


8. Start the **ReleaseFromCage** command to detach the fork from the control object.
9. Select the control object. From the **Edit** menu, pick **Explode**.
   The **Explode** command will change a control object into normal geometry.

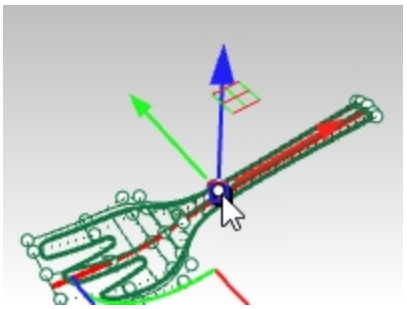
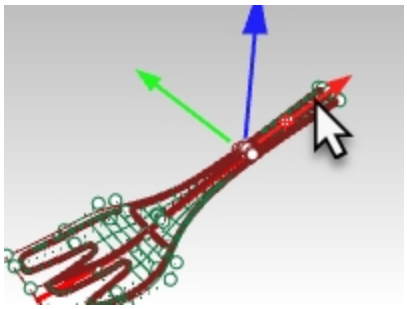


10. From the **Surface** menu, click Extend Surface (Extendsrf).
11. Pick on the long edges of the surface (Type=Smooth) at a factor of 5.
    The surface now extends past all of the fork object.

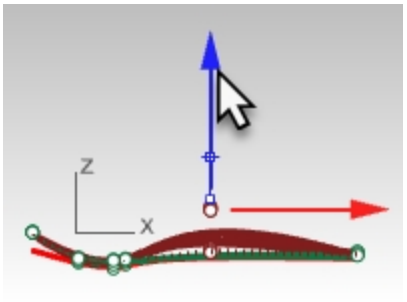
12. Start **CageEdit** again and select this extended surface as the control object. CageEditing should now result in a clean object.

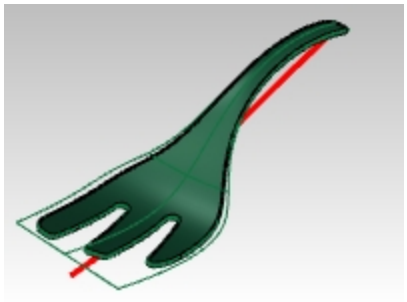



13. For example, select the two control points that are in the middle of the fork part of the object. Using Gumball, move these down the length of the fork, to increase the amount of dish in the fork.

14. Move the group of points that is at the narrowest part of the handle up or down.
This will increase or decrease the curve in the handle.



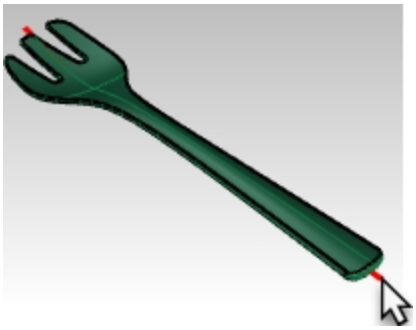


CageEdit with a curve from the object

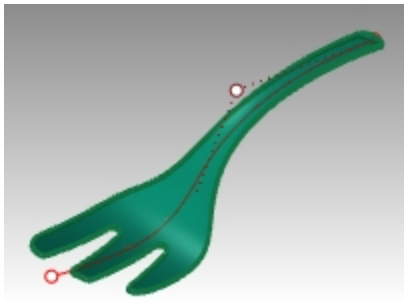
A curve can be used as a control object as well. In many cases it may make sense to use a curve which already has the same basic shape as the object. You can draw such a curve or it may be useful to extract an isocurve from the object itself and use this as the control.

1. Re-open the model **Salad_Cage.3dm** without saving.
The red curve is an extracted isocurve from the bottom surface that has been extended slightly at each end with the **Extend** command.
The reasons for extending the control curve out beyond the geometry are the same reasons as were described above for the surface control object.
Primarily, this is done to limit the distortion in the resulting polysurface.

2. Select the salad fork and start **CageEdit**.
3. Select the red curve as the **Control object**.



4. Edit the shape of the fork by point editing the control object.



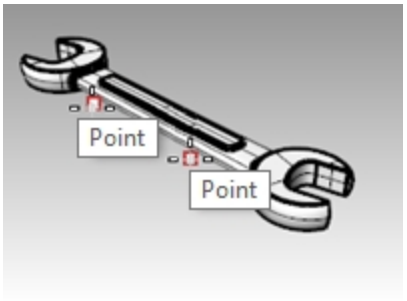
Using other deformation tools

Stretch

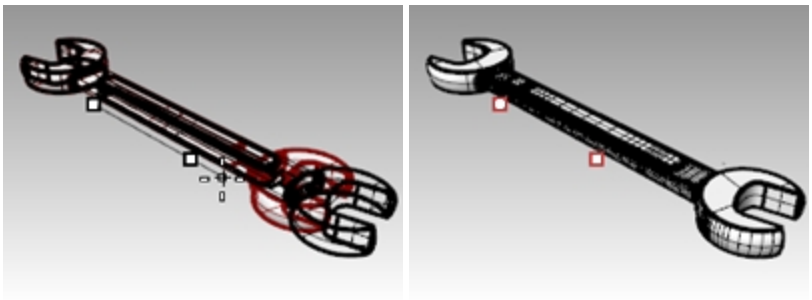
The **Stretch** command allows you to scale a selected area of an object in one direction.

Exercise 14-3 Stretch an object

1. **Open** the model **StretchWrench.3dm**.
2. **Select** the wrench.
3. Start the **Stretch** command *(Transform menu: Stretch)*.
4. For the **Start** and **End of axis**, snap to the two locked points.



5. For the **Point to stretch to**, pull the cursor to one side or another to stretch or compress the wrench.
   The section of the wrench that falls between the points that set the stretch axis is the part that is deformed. The object outside of this initial axis is moved, but not deformed. The shape of working parts of the wrench at either end are not affected, but the overall object is made longer or shorter.



Orient an object on a surface

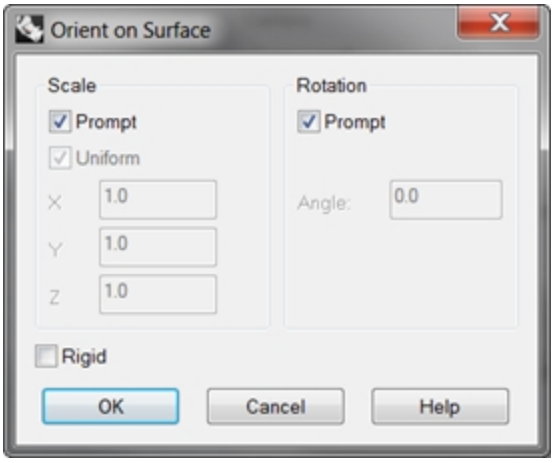
Here the goal is to place the small detail that is off to the side of the cup onto the cup handle. The detail is easy enough to build in a flat, orthographic orientation but would be quite tedious to build accurately in place on a curved surface.

The **OrientOnSrf** command can place an object on an arbitrarily curved and oriented surface with good control over the placement and can optionally also deform the object to match the curvature of the target surface.

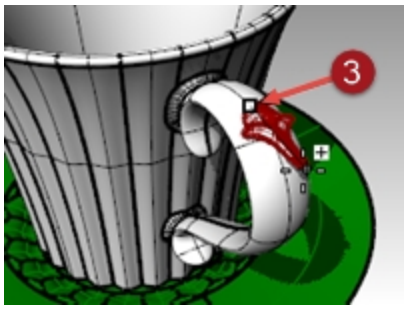
Exercise 14-4 Place a small detail on an object

1. Open the model **OrientOnSrf_detail.3dm**.
2. Start **OrientOnSrf**, and select the detail object.
3. For the **Base point**, snap to the point marked 1.
   This will be the point that is placed on the target surface.
4. For the **Reference point**, snap to the point marked 2.
   This point and the line between it and the base point will be used to set scaling and orientation on the target surface. The current **CPlane Z direction** will be mapped to the target surface normal.

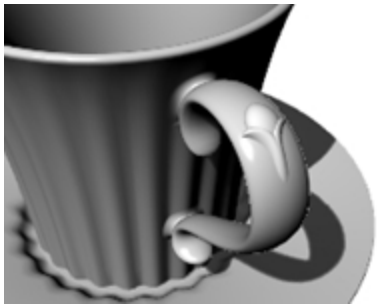
5. The **Surface to orient on** is the handle surface of the cup.
6. In the dialog box, make sure the **Rigid** option is not checked.
   This will allow the object to deform against the target surface.
7. Set **Scale** and **Rotation** to **Prompt** so that you can set these interactively as needed.
   Rotation defaults to setting the line between the base and reference points set above to the U direction of the target surface.
8. Click OK.



9. For the **Point on surface to orient to**, snap to the point on the handle marked **3**.
   Notice that in the preview the location corresponding to the base point is what tracks on the target surface.
   With the command-line **Copy=Yes** option, you can place the object without disturbing the flat original and you can place multiple copies.
10. Set **Copy=Yes**.
    There are other command-line options for **Flip**, in case the object maps to the wrong side of the surface, and **IgnoreTrims**.
    If **IgnoreTrims** is set to **Yes**, the object can be placed anywhere on the underlying surface of a trimmed face, otherwise location is restricted to the trimmed face.



11. Click to set the location point.
    Since **Scale** was set to **Prompt**, you can drag interactively to scale the object or you can type in a scale factor at the command-line.
    In this case, a **Scale factor** of **.7** works well.
12. You can rotate the object on its base point.
    Holding **Ortho** can lock the angle to match the construction plane, or you can type in an angle at the command-line. In this case, the **Rotation angle** is **zero**.
    After setting the angle, the object is mapped to the target surface.
13. At this stage, you can continue to add details to the handle, or press Enter to end the command.

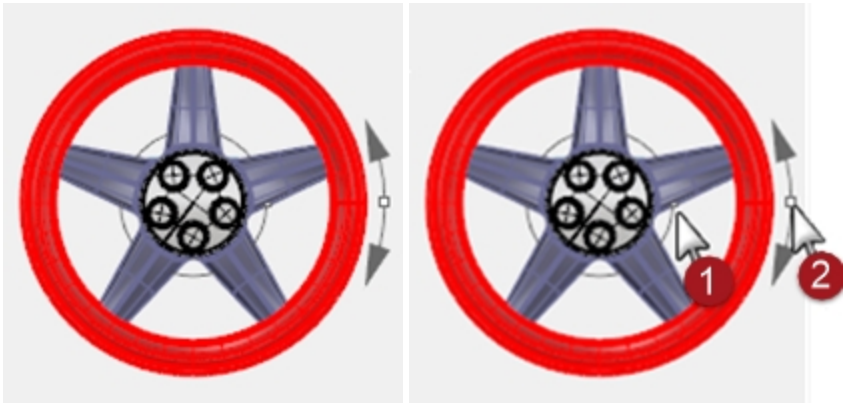


Deform an object in a spiral

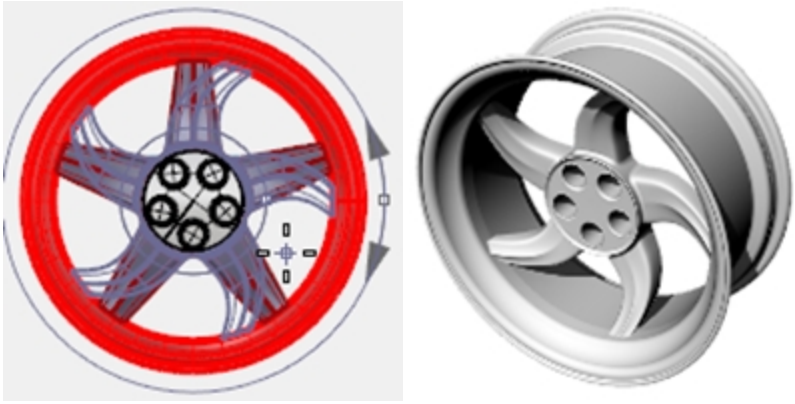
In this example, we will deform the spokes of a wheel to give them a twisted look with the Maelstrom command.

Exercise 14-5 Deform using Maelstrom

1. Open the model **Maelstrom.3dm**.
2. Select the spokes of the wheel as the objects to deform and start the **Maelstrom** command *(Deformation Tools toolbar)*.
3. In the **Front** viewport, set the **Center of Maelstrom** at **0,0,0**.
4. For the **Radius**, snap to the point on the locked circle near the center of the wheel (1).
5. For the **Second radius** snap to the point to the right of the wheel on the arc (2).



6. For the **Coil angle**, drag the cursor around the circle about 15 degrees to deform the spokes into a soft spiral shape.



Options:

- Copy = Yes/No
- Rigid = Yes/No If Yes, objects will be rotated and moved along the spiral but will not be deformed themselves. This is useful for 'Maelstroming' groups and patterns of discreet objects.

Flow along a curve

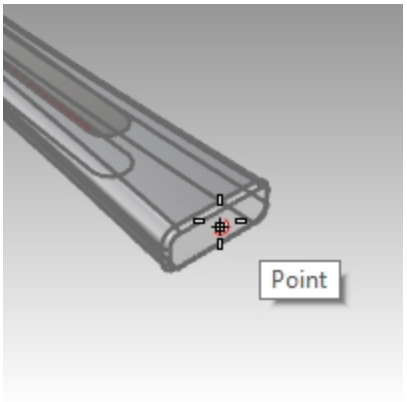
In this example, we will flow an object along a curve while recording history. This will allow us to do a secondary deformation of the original object and see the update on the flowed piece.

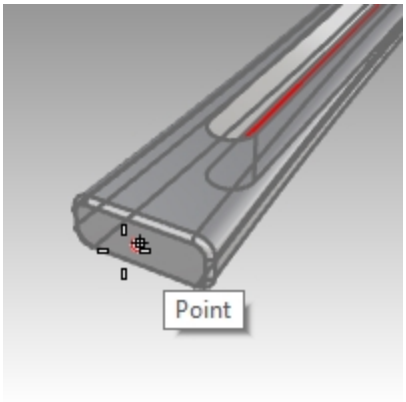
Exercise 14-6 Deform an object by flowing it along a curve

1. Open the model **Flow & Twist.3dm**.
2. On the **Status bar**, turn **Record History** on.
3. Right-click the **Record History** pane, and check the **Update Children** option. (This should already be checked by default)

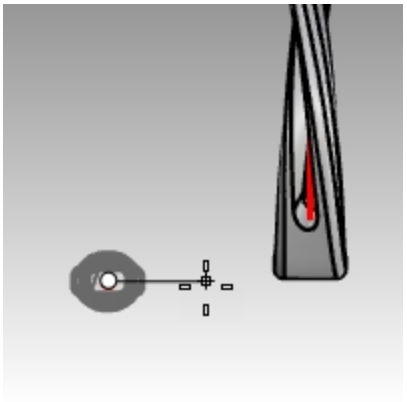
4. Select the polysurface and from the **Transform** menu, click **Flow along Curve**.
   Set **Copy=Yes**, **Rigid=No**, **Local=No**, and **Stretch=Yes**.
5. For the **Base curve**, select the straight line (through the center of the polysurface).
6. For the **Target curve**, select the circle.
   A copy of the polysurface flows around the circle.

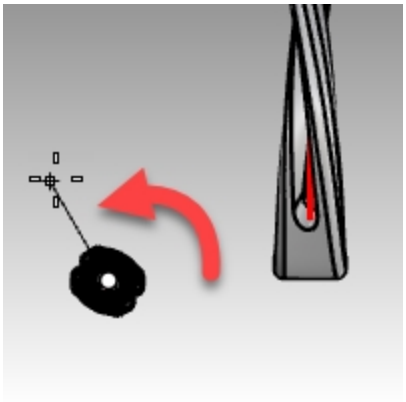
7. **Twist** the original polysurface 360 degrees or more.
   Select the original polysurface, and from the **Transform** menu, click the **Twist** command.
8. In the **Perspective** viewport, pick the points at the end of the part as the twist axis.





9. Move the cursor in a counter clock-wise movement to show the final angle 360 degrees or more and pick to preform the Twist.





The polysurface created with the **Flow** command and **History** updates to match the twisted polysurface.

Flow

The **Flow** command re-aligns an object or group of objects from a base curve to a target curve.

Steps

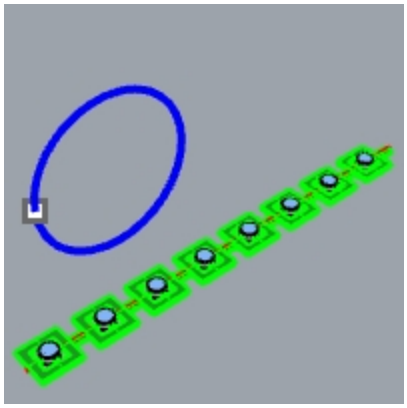
1. Select objects.
2. Select the base curve near one end.
3. Select the target curve near the matching end.

Similar to **Flow Along Surface**, the **Flow** command lets you flow solids along a curve. This makes designing in 3-D easier and lets Rhino do all the morphing work.

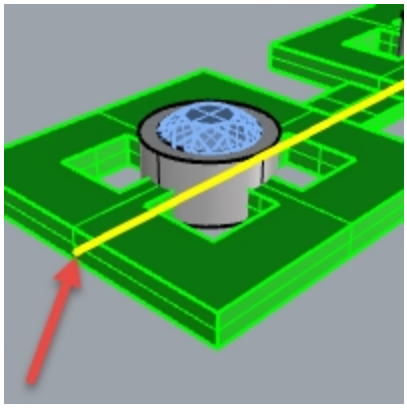
Making a Ring with Flow

Exercise 14-7 Flow the parts of a ring along the shank curve

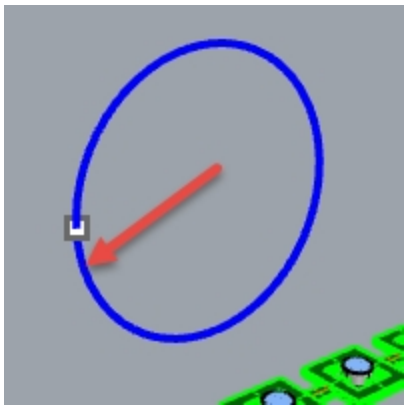
1. **Open** the model **Flow_ring.3dm**.
2. Select the green polysurface as the object to flow.
3. On the **Transform** menu, click **Flow along Curve**.



4. For the **Base curve**, select the red linear curve towards the left end.
   Note: In the **Perspective** viewport, change the display mode to **Ghosted** to view and select the base curve more easily.

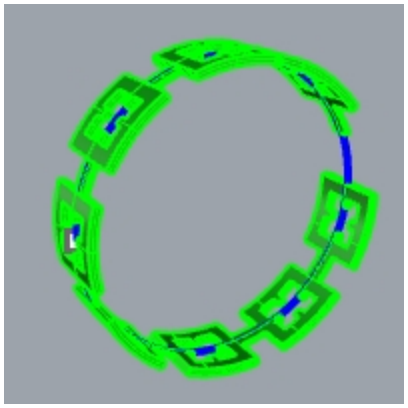


   Base curve.
5. Stop at this stage and confirm the following option settings in the command-line **(Copy=Yes Rigid=No Stretch=No)**.
6. As the **Target curve**, select the circle curve slightly below the point location.



The polysurface is morphed or flowed to the shape of the target curve.
Notice that the polysurface doesn't flow completely around the circle.

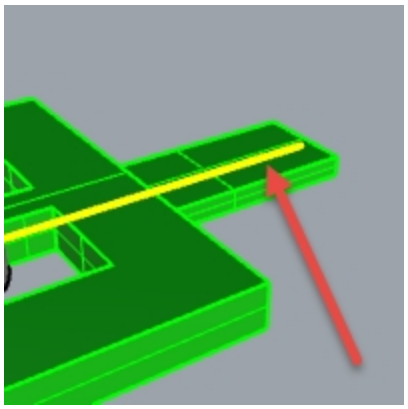
7. **Undo**.



You will flow this polysurface a few more times and use different options.
First, you will change the direction of the Flow.

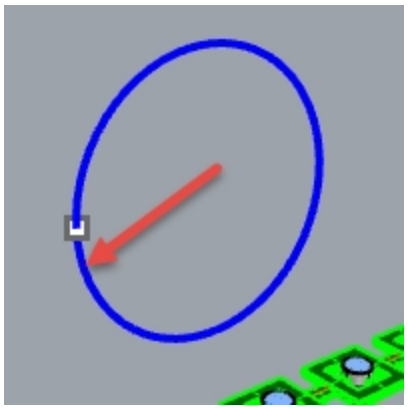
Flow the parts of a ring along the shank curve in a different direction

1. Repeat **Flow along curve** with identical steps, except select the **Base Curve** at the opposite end.



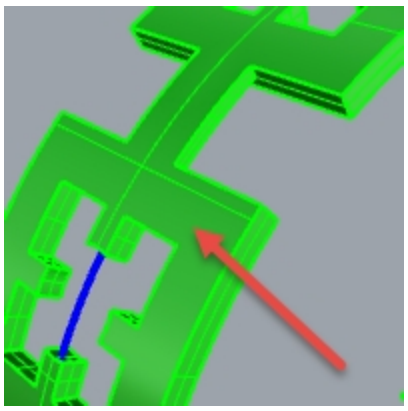
Base curve.

2. As the **Target curve**, select the circle curve slightly below the point location.

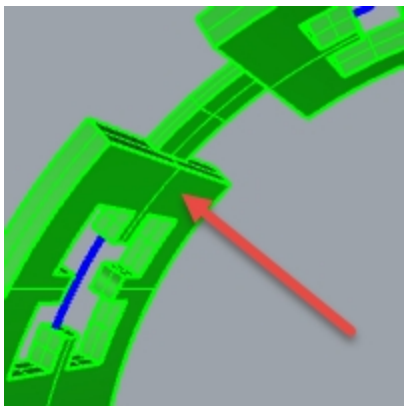


Notice that the inside and outside of the original polysurface has flipped.

3. **Undo** again.



Bottom of original polysurface is on the outside.

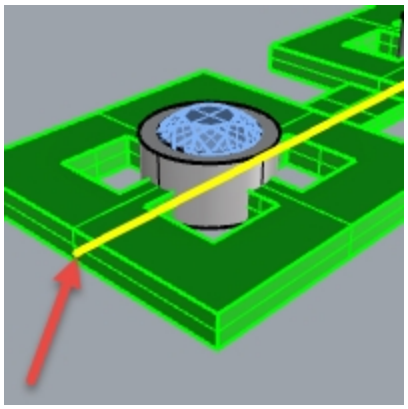


Top of original polysurface is on the inside.

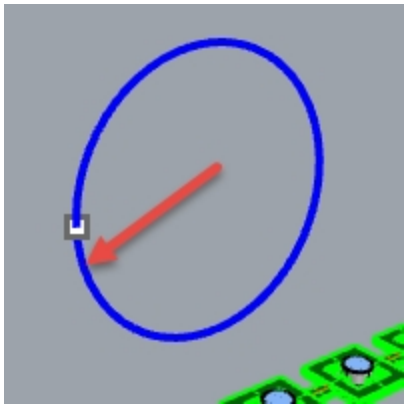
Second, you will stretch the original polysurface so that it fits completely around the circle.

Flow the parts of a ring along the shank curve, stretching it to fit the whole curve

1. Repeat **Flow along curve** the same way you did it the first time, selecting the **Base Curve** near the left end.
   Note: In the **Perspective** viewport, change the display mode to **Ghosted** to view and select the base curve more easily.



2. Stop at this stage and confirm the following option settings in the command-line **(Copy=Yes Rigid=No Stretch=Yes)**.
3. Select the circle curve slightly below the point location as the **Target curve**.

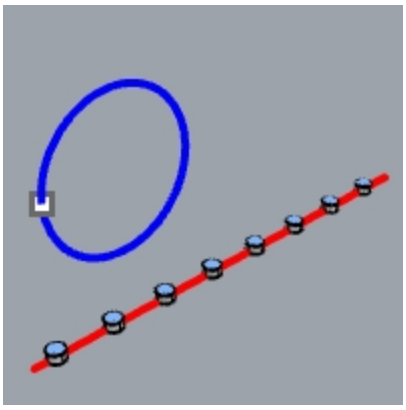


The polysurface is morphed or flowed completely around the circular shape of the target curve.

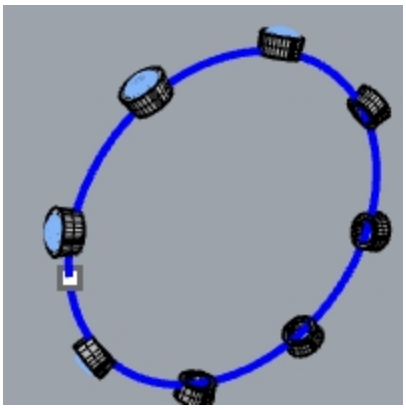
4. Use the **What** command to confirm that it is a closed solid polysurface.

Flow the gems and bezels

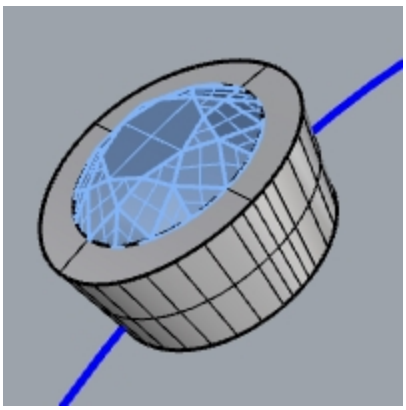
1. Hide both the original polysurface and the flowed polysurface.
2. On the **Transform** menu, click **Flow along Curve**.
3. For the **Objects to flow**, you will select the gems and bezels by layer.



4. In the **Layers** panel, Right click on the **Bezel** layer. Pick **Select objects** from the cursor menu.
5. In the **Layers** panel, right click on the **Gem_ruby**layer. Pick **Select objects** from the cursor menu.
6. Enter to close the object selection.
7. Next, select the **Base Curve** near the left end.
8. Stop at this stage and confirm the following option settings in the command-line:
   (Copy=Yes Rigid=No Stretch=Yes).
9. As the **Target curve**, select the circle curve slightly below the point location.
   The bezels and gems are morphed to fit around the circle.

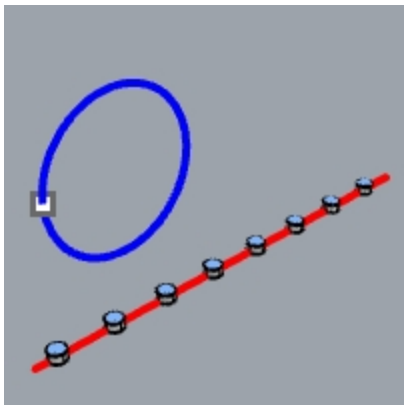


10. Examine the results.
    The sides of the bezels are not perpendicular, the top surface is not flat, and gem is stretched.
11. **Undo**.

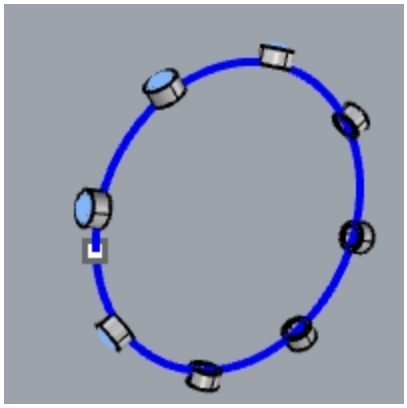


To flow the gems and bezels with Rigid=Yes

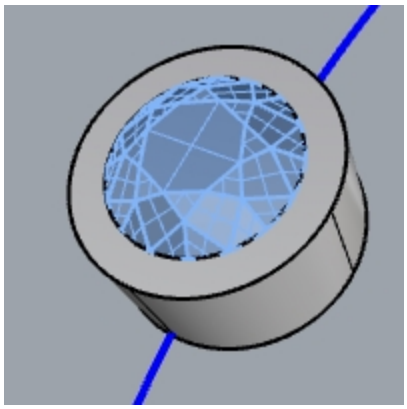
1. On the **Transform** menu, click **Flow along Curve**.
2. For the **Objects to flow**, select the gems and bezels in the **Layers** panel.
   In the **Layers** panel, right-click on the **Bezel** layer. Pick **Select objects** from the cursor menu.
   In the **Layers** panel, right-click on the **Gem_ruby** layer. Pick **Select objects** from the cursor menu.
3. Enter to close the object selection.



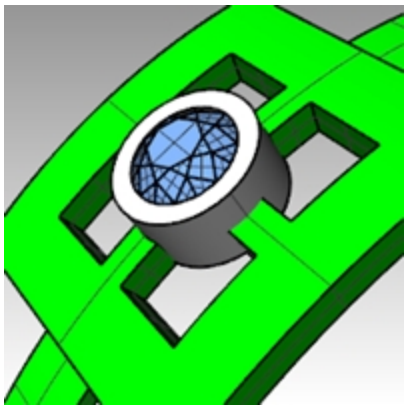
4. Select the **Base Curve** towards the left end.
5. Stop at this stage and confirm the following option settings in the command-line:
   (Copy=Yes Rigid=Yes Stretch=Yes).
6. As the **Target curve**, select the circle curve slightly below the point location.
   The bezels and gems are stretched to fit around the circle, but the objects are not deformed.



7. Examine the results.
   The sides of the bezels are perpendicular, the top surface is flat, and gem is not stretched.



8. **Show** the green polysurface again.



Chapter 15 - Blocks

There are two main advantages to using blocks in Rhino:

- Identical objects can be edited or replaced at the same time.
- Since many identical objects refer to one definition, files with many repeated objects can be smaller, sometimes much smaller, than files where each of those is its own independently defined object.

In Rhino, a block appears as a collection of objects. These can be simple 2-D or complex 3-D objects. A block can be composed of lines, polylines, free-form curves, surfaces, polysurfaces, solids, dimensions, text, and even other blocks. When a block contains other blocks, these are said to be nested. The level of nesting that may be used is not limited.

Instances and definitions

Each block has a single definition. The definition is the set of objects that comprise the block. This definition is hidden from the user. Its purpose is to provide the definition for the instances of the block that appear in the model. There can be any number of instances of a block, but only one definition. Thus, when the user modifies a block definition, all of the instances of that block will reflect the changes.

Defining blocks

Blocks can be defined using the **Block** command. This adds a new block definition and leaves one instance of the block in place.

Block instances can be added by copying existing instances, or by using the **Insert** command. Insert allows you to choose from a list of existing block definitions, or to browse to an external file.

Note: There may be block definitions in a file for which no instances appear. Deleting an instance does nothing to the definition.

Insertion points

Each block has an insertion point. This is a base location for the block as a whole and is in fact the point used when a block instance is added using the **Insert** command.

Embedded and linked blocks

When the definition of a block is saved in the Rhino file, the block is said to be embedded. If the block definition exists as a separate file, the block is linked. In the latter case, saved changes to the external file are what govern the appearance of the block instances in all of the files into which it is inserted.

Layers and blocks

Layers can be confusing when using blocks. It is important to keep in mind that any instance of a block will exist on some layer just like any Rhino object.

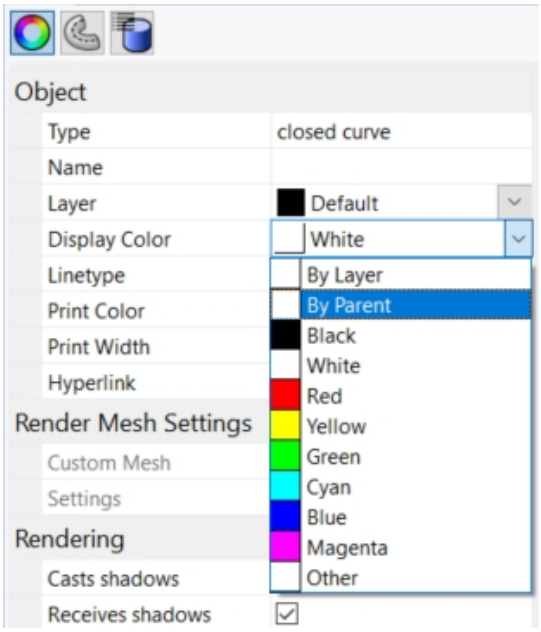
- Typically, this is the layer that is current when the instance is inserted, but the instance layer can be changed, just like any other type of object. However, each object in the block definition also exists on some layer, independent of the block instance as a whole.
  For example, two instances of the same block can be on different layers, but the objects comprising the block definition will be on the same layers in each instance.
- Turning on or off a layer containing an instance will show or hide the entire instance. Turning on or off layers of the objects contained in a block will show or hide that part in every instance of the block, regardless of what layer the instance is on.
- Linked blocks' layers (reference layers) can be shown in the layer panel as normal layers, or displayed as special reference layers.

The Block Rules

Here are some rules that are helpful to keep in mind when using blocks.
It is best to read these over until they make sense to you.
Also, you will experiment with blocks in this chapter. You will become more familiar with the rules just by creating and inserting the block objects.

- The layer that is current when the block instance is inserted is called the reference layer. The inserted instance resides on this layer.

- The visibility of the entire block is controlled with the block reference layer.
- If you turn the block reference layer off, the entire block will not be visible, regards less of the layers to which the geometry is assigned.
- Objects in the block can be assigned certain properties with the "by parent" option. This give the objects a unique behavior when grouped together in a block.
- Object properties of display color, plot color, and plot width can be assigned "by parent."



- Objects assigned "by parent" and grouped into a block have a chameleon-like property. They change to display the color and the plot weight of the block's reference layer.
- This powerful feature allows a block to look different, without creating a separate block.
- The layer that is current when the block is made does not have any bearing on the block definition itself.
- The layers of the objects in the block definition, and the layer that is current when an instance is inserted are important in how the block instances behave with respect to visibility and appearance.

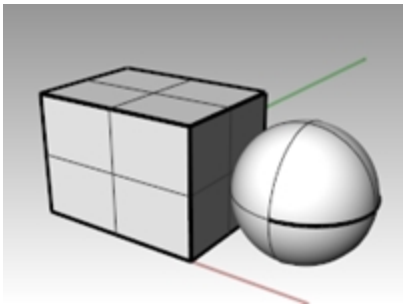
Blocks

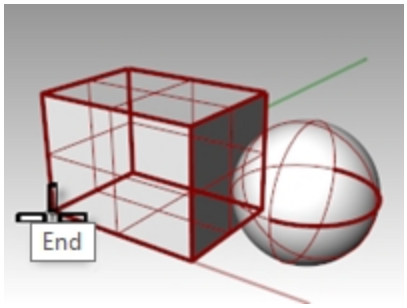
Block definitions are most easily edited using the **BlockEdit** command, or by double-clicking on a block instance. If the block is a linked block, editing will open the linked file in a new instance of Rhino and the current Rhino will be suspended until that second Rhino is closed.

Exercise 15-1 Block basics

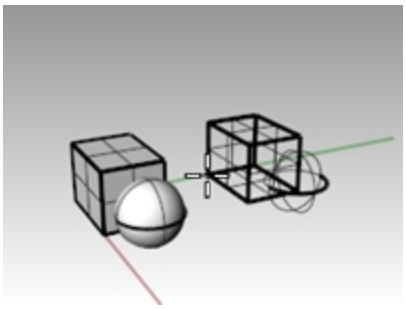
Make a block

1. Start a new model.
2. Draw a **box** and a **sphere** somewhere near the origin.
3. **Select** the two objects
4. Use the **Block** command *(Edit menu: Blocks > Create Block Definition)* to make a block.
5. For the **Block Base Point**, snap to a corner of the box.
   The point you select will become the insertion point for the block.

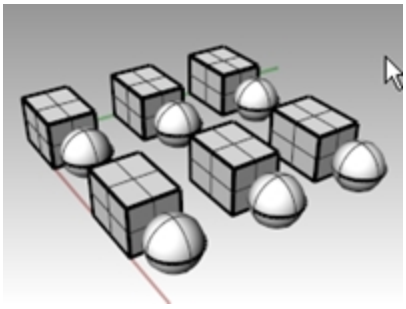




6. In the **Block Definition Properties** dialog box, **Name** field, type **Test 1**, click OK.
7. Use the **Insert** command *(File menu: Insert)* to insert the new block.
8. In the drop-down list at the top of the **Insert** dialog box, select **Test 1**.
   Make sure to insert as **Block Instance** and not as a **Group** or **Individual Objects**.
9. Accept the defaults for **Scale** and **Rotation**.
10. Place the block in the Rhino scene.
    Notice that the cursor follows the location that you set as the block base point when the block was created. This is the insertion point.

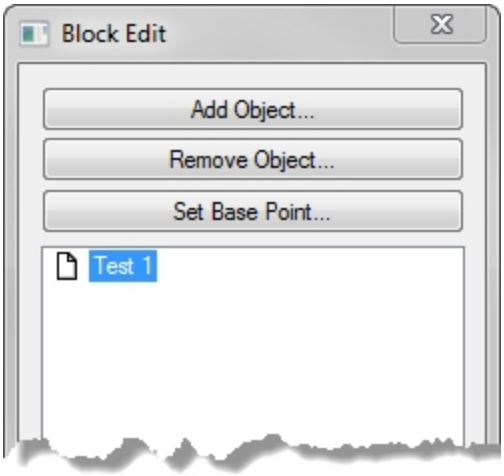


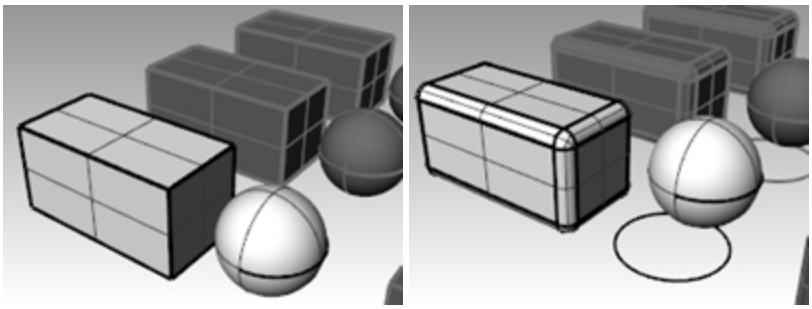
11. **Select** the block instance and make one or two more copies of this instance using the **Copy** command.



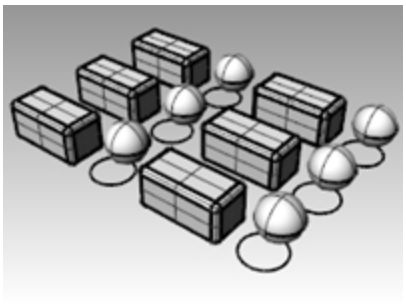
Redefine a block

1. **Double-click** one of the block instances.
2. This opens the **Block Edit** dialog box, returns the original geometry at the block location, and turns all of the other blocks to a dark shaded color.
   The sphere and box now select individually.
3. Use **FilletEdge** to fillet the edges of the box, **Move** the sphere slightly, and add a **Circle**.





4. Click OK on the **Block Edit** dialog box.
   Notice the other instances of the block placed and copied earlier are now updated and look like the redefined block. Instead of a box and a sphere, the blocks have a filleted box, a moved sphere, and a circle.

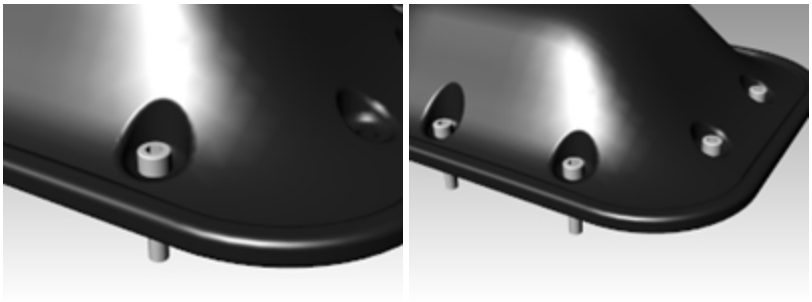


Files as blocks

The **Insert** command has options for insertion point, scale, and rotation. The block can be inserted as a block instance, a group, or individual objects.

Exercise 15-2 Inserting files as block

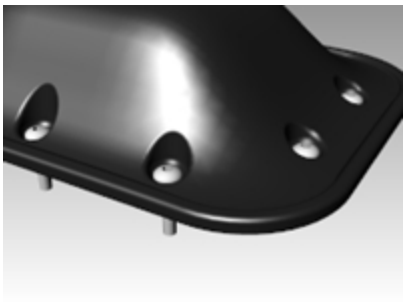
1. Open the **Blocks-mm.3dm** model.
2. Make the **Fasteners** layer current.
3. Use the **Insert** command *(File menu: Insert)* to insert the **FILH-M6-1.0-25.3dm** model.
4. In the **Insert** dialog box, choose **Insert as Block Instance**, and click OK.
5. In the **Insert File Options** dialog box, choose **Embed and link**, click OK.
6. For the **Insertion point**, snap to the center of one of the holes in the cover.
7. **Copy** the cap screw around to all of the other holes.



Change the block

1. Start the **Block Manager** command *(Edit menu: Blocks > Block Manager)*.
2. Select the block definition for the cap screw that you inserted.
3. Click the **Properties** button.
4. In the **Block Definition Name** box, type **Fastener**.
5. Under **File name**, choose **Browse**, select **RH-M6-1.0-25.3dm**, and click Open.
6. In the **Block Definition Properties** dialog box, click OK.
7. In the **Block Manager**, click Update.
   The cap screws change to the round head cap screws and the color changes to match the layer color of the insertion layer.
   Note: Even on a small file like this, the size difference can be significant. If this file would have had the cap screws

imported and then copied around, it would be 35-40 percent larger than it is with block instances. The use of blocks can help to reduce problems caused by large file sizes.



Chapter 16 - Troubleshooting

The troubleshooting tools are most often used for repairing files imported from other programs.

Some Rhino operations can make "bad objects" under certain circumstances. Bad objects may cause failure of commands, shade and render badly, or export incorrectly.

It is good practice to use the **Check** *(Analyze menu: Diagnostics > Check)* or **SelBadObjects** *(Analyze menu: Diagnostics > Select Bad Objects)* commands frequently during modeling. If errors can be caught right away the objects can often be fixed more easily than if the bad part is used to make other objects.

If the goal is to create a rendering or a polygon mesh object, some errors can safely be ignored so long as they do not get in the way of building the model itself in later stages.

For objects that must be exported as NURBS to other applications for further engineering or manufacturing, it is best to eliminate as many errors as possible.

General strategy

The troubleshooting steps will be the same, whether or not the file was created in Rhino or another application. Over time, you will discover patterns of problems and develop procedures to fix them.

Although the techniques used vary greatly depending on the individual file, we will focus on a general strategy for repairing problem files

Start with a clean file

When possible, spending a little time in the originating application to export a "clean" file will save a great deal of cleanup work later. Unfortunately, this is not always an option.

Guidelines for repairing files

1. **Hide** or **delete** extra data.
2. Use the **SelDup** command *(Edit menu: Select Objects > Duplicate Objects)* to find duplicate entities and delete them or move them to a duplicate layer in case you need them later.
3. **Hide** curves and points.
4. Use the **SelSrf** command *(Edit menu: Select Objects > Surfaces)* to select all the surfaces or the **SelPolysrf** command *(Edit menu: Select Objects > Polysurfaces)* to select all the polysurfaces.
5. **Invert** *(Edit menu: Select Objects > Invert)* the selection, and move the selected items to another layer and turn it off.
   This will leave only surfaces or polysurfaces on the screen.
6. Check for bad surfaces.
   The **Check** and **SelBadObjects** commands will determine if some of the surfaces in the model have problems in their data structures. Move these surfaces to a "bad surfaces" layer for later clean up.
   If the bad object is a polysurface, use the **ExtractBadSrf** command to extract the bad surfaces from the original polysurface.
   Then you can fix the bad surfaces and then use the **Join** command to reattach them to the good part of the polysurface.
7. **Shade** the viewport and visually inspect the model.
   Does it look like you expected it would?
   Are there obviously missing surfaces?
   Do surfaces extend beyond where they should?
   The trimming curves needed to fix them may be on the duplicate layer.
8. Look at the **Absolute tolerance** setting in the **Document Properties** dialog box on the **Units** page.
   Is it reasonable? Free-form surface modeling requires an intelligent compromise in modeling tolerance. Surface edges are fitted to neighboring surface edges within the specified modeling tolerance. The tighter the tolerance, the more complex these surfaces become and system performance suffers. There is no point in fitting edges to a tight tolerance that is not supported by your down-stream manufacturing processes or by the precision of the input data.
9. **Join** *(Edit menu: Join)* the surfaces.
   When joining, edges are joined if they fit within the specified modeling tolerance. If they are outside the tolerance, they are not joined. Joining does not alter the geometry. It only tags the edges as being close enough

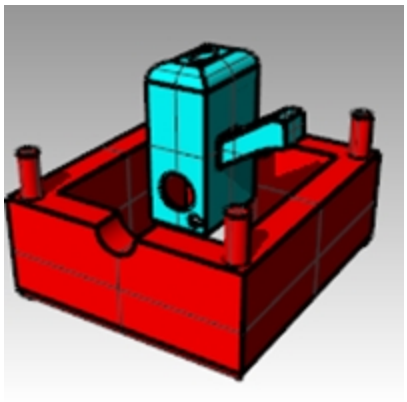
to be treated as coincident, and then one edge is discarded.
Look at the results on the command-line. Did you get as many polysurfaces as you thought you would? Sometimes there are double surfaces after importing an IGES file. Usually, one will be complete and the second one will be missing interior trims. When the Join happens, you have no control over which of the two surfaces it will select. If you suspect this has occurred, try joining two naked edges. If there is no nearby naked edge where one should be, Undo the Join, and select for duplicate surfaces. Delete the less complete surfaces and try the Join again.

10. Check for **naked edges**.
    Naked edges are surface edges that are not joined to another surface. During the **Join** process, the two edges were farther apart than the specified modeling tolerance. This may be from sloppy initial modeling, a misleading tolerance setting in the imported IGES file, or duplicate surfaces. If there are too many naked edges showing when you run the **ShowEdges** command *(Analyze menu: Edge Tools > Show Edges)*, consider undoing the **Join** and relaxing the absolute tolerance and try the **Join** again. It is likely that the original modeling was done to a more relaxed tolerance and then exported to a tighter tolerance.
    Note: You cannot improve the tolerance fitting between surfaces without substantial remodeling.
11. **Join** naked edges or remodel.
    Joining naked edges can be a mixed blessing. It is a tradeoff and may cause problems down-stream. If your reason for joining the edges is for later import into a solid modeler as a solid, or a meshing operation like making an STL file, using the **JoinEdge** command *(Analyze menu: Edge Tools > Join 2 Naked Edges)* will not generally cause any problems. If you will be cutting sections and most other "curve harvesting" operations, the sections will have gaps as they cross edges that were joined outside of tolerance. The gap to be spanned is displayed prior to joining. If the gap is less than twice your tolerance setting, you can proceed without worry. If the gap is too wide, consider editing or rebuilding the surfaces to reduce the gap. Join and JoinEdge do not alter the surface geometry. They only tag edges as being coincident within the specified tolerance.
12. **Repair** the bad surfaces.
    It is best to repair one bad surface at a time, and Join them into the polysurface as you go. In order of least destructive method to most radical, the problems that caused them to fail **Check** can be repaired by the following:
    - **Rebuild** edges.
    - **Detach trim curves** and re-trim.
    - **Rebuild surfaces** (surfaces change shape).
    - **Replace surfaces** - harvest edges from surrounding surfaces, cut sections through bad surfaces and build replacement surfaces from the collected curves.
13. **Check** for **bad objects**.
    Sometimes joining surfaces that pass check can result in a polysurface that fails check. Generally, this is caused by tiny segments in the edge or trimming curves that are shorter than the modeling tolerance.
14. Extract the adjoining surfaces, check them, use the **MergeEdge** command *(Analyze menu: Edge Tools > Merge Edge)* to eliminate these tiny segments, and join them back in.
    You are finished when you have a closed polysurface that passes **Check** and has no naked edges. As you are joining and fixing surfaces, it is generally a good idea to run **Check** from time to time as you work.
15. **Export**.
    Now that the model has been cleaned up and repaired, you can export it as IGES, Parasolid, or STEP for import into your application.

Exercise 16-1 To try these procedures

Try these procedures

1. **Open** the model **Check 01.3dm**.
   This file has a bad object.
2. Find the bad object, fix it, retrim, and rejoin the surfaces.
3. **Open** the file **Check 02.igs**.
   This file has several problems. It is representative of commonly found problems with IGES files.
4. After repairing the bad object and trimming it, look for other objects that do not appear to be trimmed correctly.



Chapter 17 - Polygon meshes

In Rhino, a polygon mesh is a collection of vertices, edges and faces. A mesh can be used to approximate surface and solids in Rhino. The mesh is generated from the Rhino model for fabrication like CNC and 3D printing. The mesh is also used for rendering and analysis like Gaussian curvature, collision detection and FEA.

Although Rhino is a primarily a NURBS surface modeler, some tools are included to create and edit polygon mesh objects.

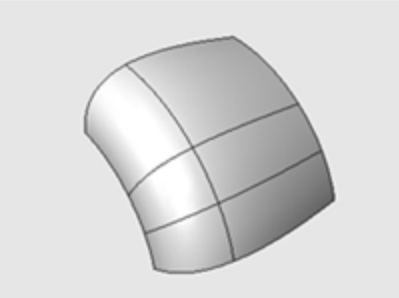
You will explore different methods for creating and editing meshes for different purposes. Downstream requirements are the most important considerations when determining which technique to use for meshing. If the mesh is going to be used for rendering, you will use different mesh settings than you would use for a mesh that will be used for manufacturing (machining or prototyping).

Render meshes

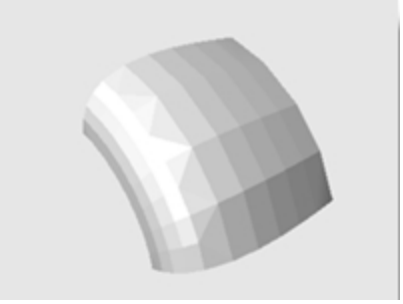
When meshing for rendering, appearance and speed are the most important considerations. You should strive to achieve a mesh with as few polygons as possible to get the look you require. The polygon count will affect performance, but too few polygons might not give you the quality you are after in the final rendering. Generally, if it looks good, then you have the right setting.

Meshes for manufacturing

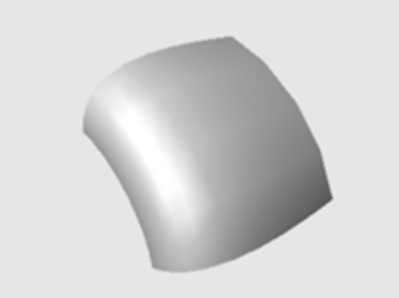
Meshing for manufacturing is an entirely different situation. You should try to achieve the smallest deviation of the mesh from the NURBS surface. The mesh approximates the NURBS surface and deviation from the NURBS surface may be visible in the final manufactured part.



If the mesh is not accurate enough for manufacturing, you will see visible polygon edges on your final products.

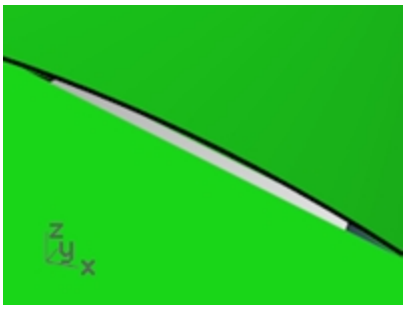


Using the same meshing setting, the rendering system can hide polygon edges and visually "smooth" the mesh to show a smooth look.

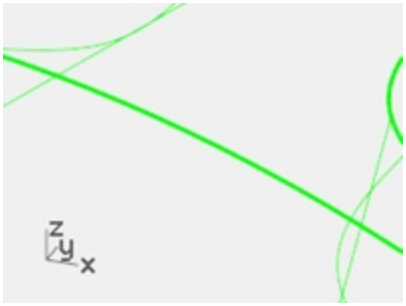


Exercise 17-1 Experiment with mesh settings

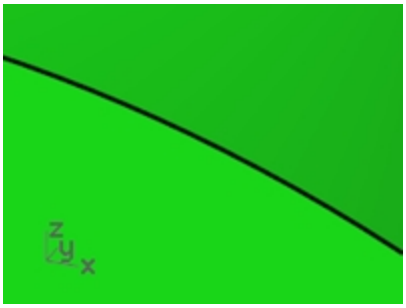
1. **Open** the model **Meshing.3dm**.
2. **Shade** the **Perspective** viewport and inspect the curved edge between the surfaces.
3. A series of angular gaps allows the background color to show.



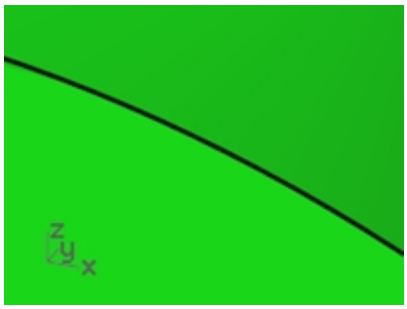
4. Get back to a wireframe view.
   The edges appear to be exactly coincident. The gaps you saw in the shaded view were due to the polygon mesh Rhino uses to create shaded and rendered views. The polygons are so coarse at the edges that they are clearly visible as individual facets.



5. In the **Document Properties** dialog box, on the **Mesh** page, click **Smooth & slower**.
6. Inspect the curved edge between the surfaces.
   The overall rounded surface is smoother and cleaner looking but the edges still have gaps.
   Although it is possible to use the Custom settings to refine the shaded mesh enough to eliminate the jagged edges, this will affect all render meshes in the model. This will increase the amount of time necessary to create meshes and may decrease the performance of shading and rendering to unacceptable levels.
7. To eliminate the gaps without refining the mesh settings, join adjacent surfaces to each other.



8. **Join** the three surfaces together.
   The mesh is refined along each side of the joined edges so that they match exactly across the edge. This eliminates the gaps visible earlier.
   Rhino saves these polygon meshes with the file in order to reduce the time needed to shade the model when it is reopened. These meshes can be very large and can increase the file size considerably.
9. On the **File** menu, click **Save Small**.
   This saves the file without the render meshes and the bitmap preview, to conserve disk file space.



Note: The meshes created by render and shading modes on NURBS surfaces and polysurfaces are invisible in wireframe display, not editable, and cannot be separated from the NURBS object. Render meshes are managed for the current model in the **Document Properties** dialog box, on the **Mesh** page. In addition, you can change per object **Render Mesh Settings** on the **Object Properties** dialog box.

Meshes from NURBS objects

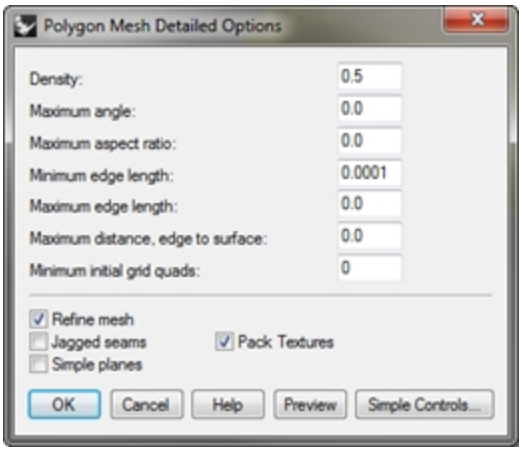
The meshes created by the **Mesh** command are visible and editable, and separate from the NURBS objects from which they were created.

Rhino has two methods for controlling mesh density: **Simple Controls** or **Detailed Controls**. With **Simple Controls**, a slider is used to roughly control the density and number of mesh polygons. With **Detailed Controls**, you can change any of seven settings and enable four check boxes to control the way the mesh is made.

The mesh is created in three steps based on the detailed criteria: initial quads, refinement, and adjustment for trim boundaries. These steps are not shown to you; it is all automatic.

In the following exercise, you learn about each of the seven detailed controls on the **Mesh** command dialog and see how they influence the mesh generation.

Polygon Mesh Detailed Options



Density

Uses a formula to control how close the polygon edges are to the original surface. Values between 0 and 1. Larger values result in a mesh with a higher polygon count.

Maximum angle

The maximum angle between adjacent faces in the mesh. Smaller values result in slower meshing, more accurate meshes, and higher polygon count.

Maximum aspect ratio

The maximum ratio length to width of triangles in the initial grid quads.

Minimum edge length

Bigger values result in faster meshing, less accurate meshes and lower polygon count. Controls the minimum length of the sides of quads and triangles of the mesh.

Maximum edge length

Smaller values result in slower meshing and higher polygon count with more equally sized polygons. When Refine mesh is selected, polygons are refined until all polygon edges are shorter than this value. This is also approximately the maximum edge length of the quads in the initial mesh grid.

Maximum distance, edge to surface

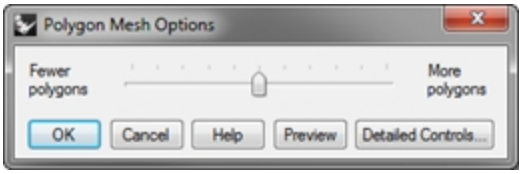
Smaller values result in slower meshing, more accurate meshes, and higher polygon count. When Refine is selected, polygons are refined until the distance from a polygon edge midpoint to the NURBS surface is smaller than this value. This is also approximately the maximum distance from polygon edge midpoints to the NURBS surface in the initial mesh grid.

Minimum initial grid quads

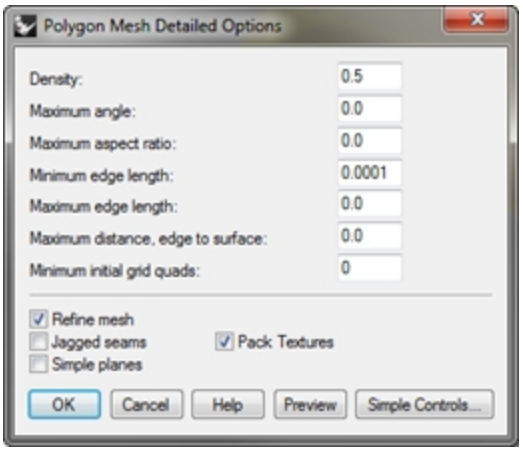
Bigger values result in slower meshing, more accurate meshes, and higher polygon count with more evenly distributed polygons. This is the minimum number of quads in the mesh before any of the other refinements are applied. If you set a number for this and set all other values to 0, this will be the mesh returned.

Create a mesh using detailed controls

1. Select the object.
2. Start the **Mesh** command *(Mesh menu: From NURBS Object)*.
   The **Polygon Mesh Options** dialog box appears.

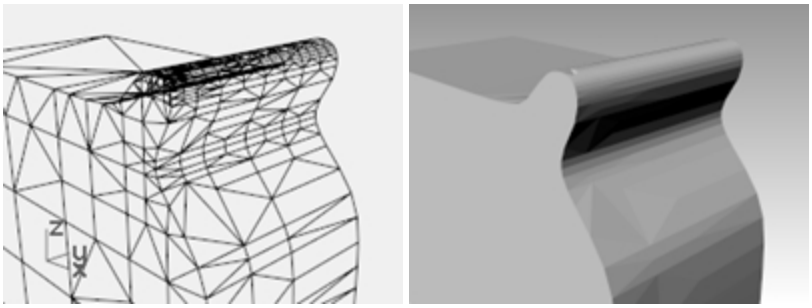


3. In the **Polygon Mesh Options** dialog box, click **Detailed Controls**.
4. The **Polygon Mesh Detailed Options** dialog box appears.
   These settings are saved to the Windows Registry when you exit Rhino.
5. In the **Polygon Mesh Detailed Options** dialog box, set the following, if they are not already set:
   Density=0.5
   Maximum angle=0.0
   Maximum aspect ratio=0.0
   Minimum edge length=0.0001
   Maximum edge length=0.0
   Maximum distance, edge to surface=0.0
   Minimum initial grid quads=0
   Check the **Refine mesh** option.
   Clear the **Jagged seams** option.
   Clear the **Simple planes** option.
   Check the **Pack textures** option.

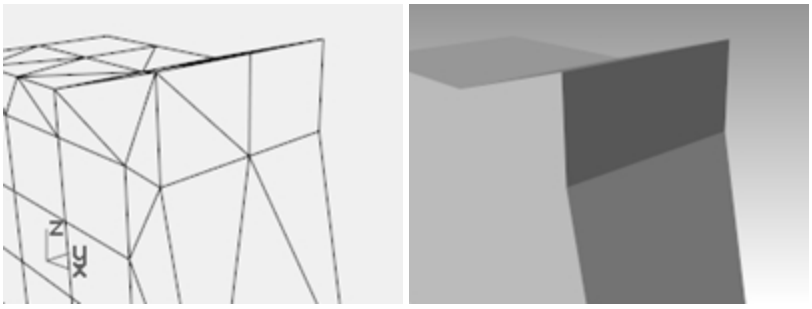


6. Click OK.
   A mesh is created using the default settings.
7. **Hide** the original polysurface, change the viewport display mode to **Rendered**, and use the **Flat Shade** display

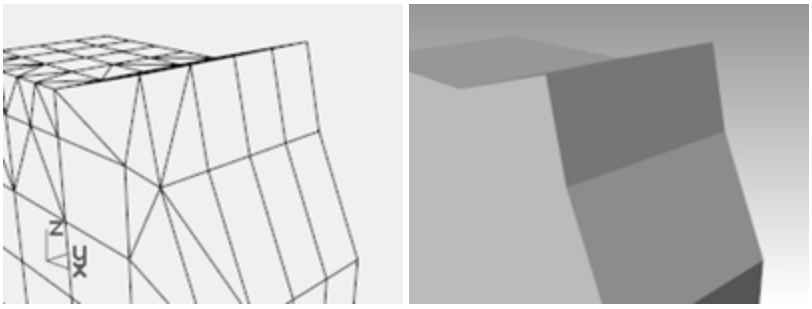
mode to view the output.
The **Flat Shade** display mode shows what the model would look like if it were output for prototyping or machining at this mesh density.



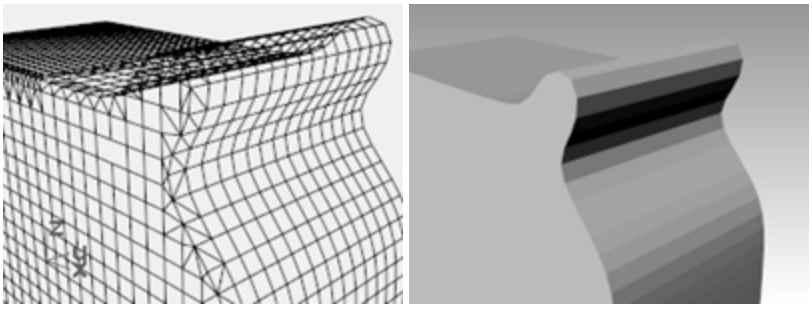
8. **Undo** the previous operation, repeat the **Mesh** command, and then make the following changes in the **Polygon Mesh Detailed Options** dialog box:
   Maximum angle=0.0
   Maximum aspect ratio=2.0
9. Click OK.
   Note the changes in polygon count, the shape of the mesh, and the quality of the flat-shaded mesh.



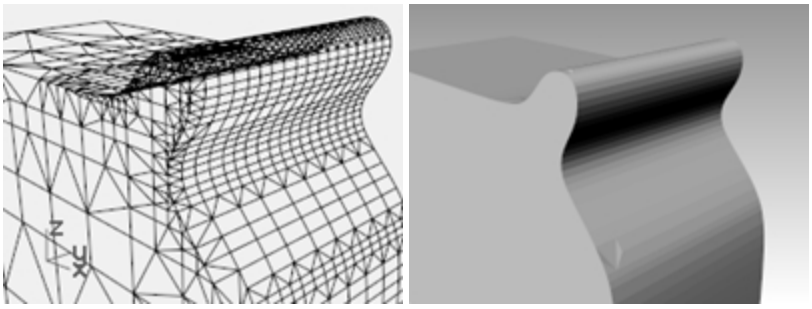
10. **Undo** the previous operation, repeat the **Mesh** command, and then make the following changes in the **Polygon Mesh Detailed Options** dialog box:
    Minimum initial grid quads=16
    Note the changes in polygon count, the shape of the mesh, and the quality of the flat-shaded mesh.



11. **Undo** the previous operation, repeat the **Mesh** command, and then make the following changes in the **Polygon Mesh Detailed Options** dialog box:
    Minimum initial grid quads=500
    Note the changes in polygon count, the shape of the mesh, and the quality of the flat-shaded mesh.



12. **Undo** the previous operation, repeat the **Mesh** command, and then make the following changes in the **Polygon Mesh Detailed Options** dialog box:
    Maximum distance, edge to surface=0.01
    Minimum initial grid quads=0
    Note the changes in polygon count, the shape of the mesh, and the quality of the flat-shaded object.



Chapter 18 - Rendering

With Rhino, creating design renderings of Rhino models is easy. Simply add materials, lights, and render.

There are several controls in Rhino's renderer that create interesting special effects in the rendering.

In the following exercise, you will render with and without isocurves, adjust colors, transparency, and ambient light to create images with special effects.

You will also look at **Environments**, **textures** and **decals** for adding realism to the rendering.

Rhino display mode **Rendered** will let you preview your materials and environment before rendering. While **Raytrace** display mode will let you work real time in the ray traced view. **Ray trace** display mode will use the **cuda cores** on your **Quadro** card or it can be configured to use the CPU.

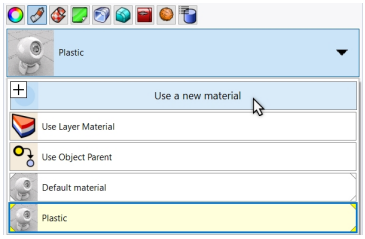
Rhino for Windows enjoys a large number of 3rd party renderers that run as plug-ins to Rhino. For more advanced rendering features, try other render plug-ins for Rhino that are available on the Food4Rhino web site.

Exercise 18-1 Rhino rendering

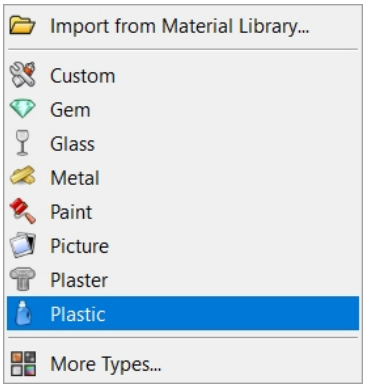
Open the model and set materials

1. On the **Render** menu, click **Current Renderer**, and then click **Rhino Render**.
2. Open the model **Finished Detergent Bottle.3dm**.

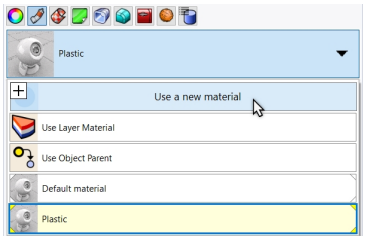
3. On the **Render** menu, click **Current Renderer**, and then click **Rhino Render**.
4. Select the bottle and on the **Properties** panel, click on the **Material** page.
5. Click the arrow next to **Use Layer Material** and pick the **Use New Material** button.



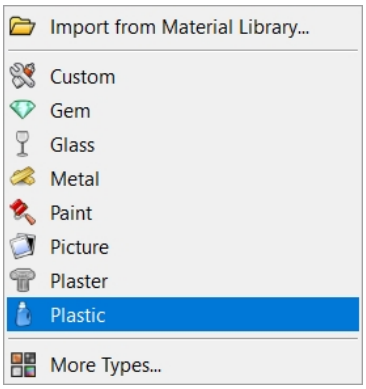
6. From the pop-up menu, pick **Plastic** material template.



7. In the new material configuration in Material panel, make the following changes:
   Name: **Light Blue**
   Color: Light Blue **(R=163, G=163, B=194)**
8. Select the cap and on the **Properties** panel, click on the **Material** page. Click the arrow next to **Use Layer Material** and pick the **Use New Material** button.

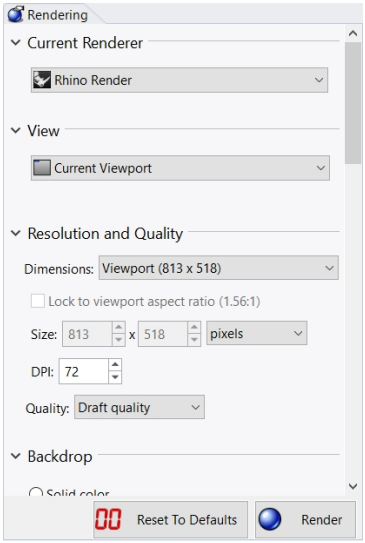


9. From the pop-up menu, pick **Plastic** material template.



10. In the new material configuration in Material panel, make the following changes:
    Name: **Tan Plastic**
    Color: Tan**(R=222, G=172, B=112)**
11. **Render** the **Perspective** viewport.

12. From the **Tools** menu , click **Options**.
13. In **Options** dialog, on the **Render** page, scroll down to the **Lighting** section, uncheck the **Skylight** option and check the **Use lights on layers that are off** option.
    Note:
    Rhino has a **Rendering** panel. The options on the **Rendering** panel are also available on the **Render** page of the **Options** dialog.
    To access the **Rendering** panel, click **Rendering** from the **Panel** menu or right-click over the **Properties** panel tab and pick **Rendering** from the pop-up menu.

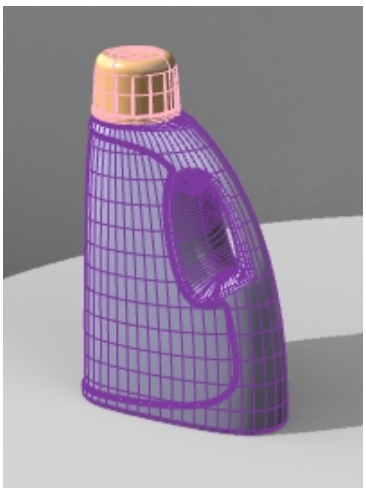


14. **Render** the **Perspective** viewport.

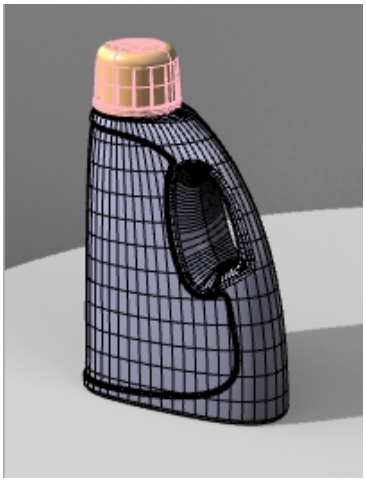
The shadow of the bottle is now appearing on the ground plane.

Render with isocurves displayed

1. From the **Tools** menu , click **Options**.
2. Under **Document Properties**, on the **Render** page, scroll down to the **Wireframe** section and check the **Render surface edges and isocurves** option.
3. **Render** the **Perspective** viewport.
   The wire color is the same as the layer color because the object's wire color is set to **By Layer**.

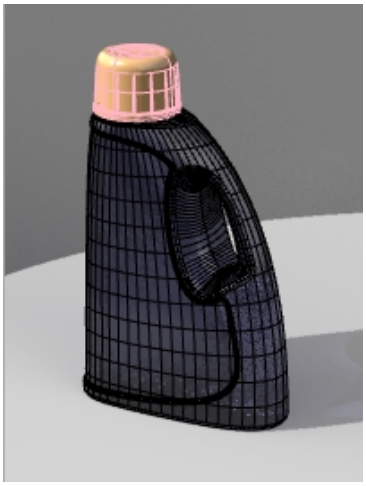


4. Select the bottle polysurface.
5. In the **Properties** panel, on the **Object** page, change the **Display Color** setting from **By Layer** to **Black.**
6. **Render** the **Perspective** viewport.
   The objects are rendered with black isocurves.

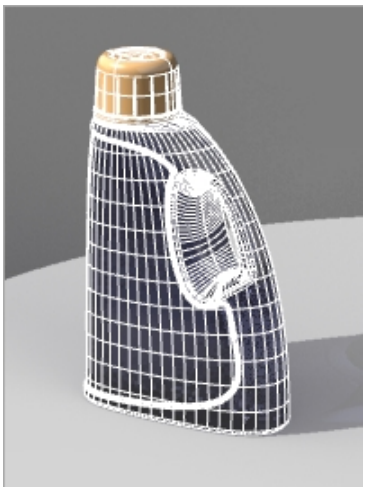


Render a transparent material with isocurves displayed

1. In the **Properties** panel, on the **Object** page, change the **Display Color** setting from **By Layer** to **Black.**
2. Next **Render** the **Perspective** viewport.
   The objects are rendered with black isocurves and the material is transparent.



3. Select the cap and the bottle polysurfaces.
4. In the **Properties** panel, on the **Object** page, change the **Display Color** setting from **By Layer** to **White.**
   The objects are rendered with white isocurves and the material is transparent.
5. Experiment with these adjustments to get the desired effect.
6. Turn on the **Lights** layer and adjust the properties of the lights for more subtle changes.



Rendering properties

With Rhino's Material Editor, you can assign any combination of color, reflectivity, transparency, highlight, multiple bitmaps, and environments.

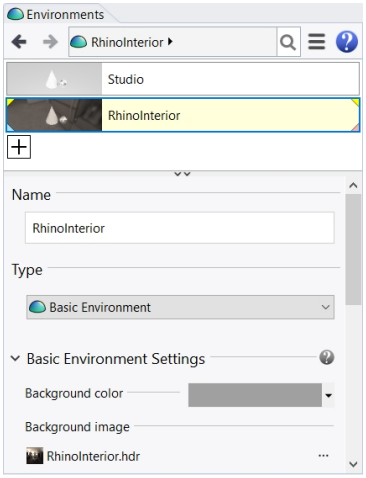
In the following exercise, we will add environment settings, add materials and lights, create custom materials, edit materials, add decals to objects, and render a scene.

Exercise 18-2 Rendering with Environments

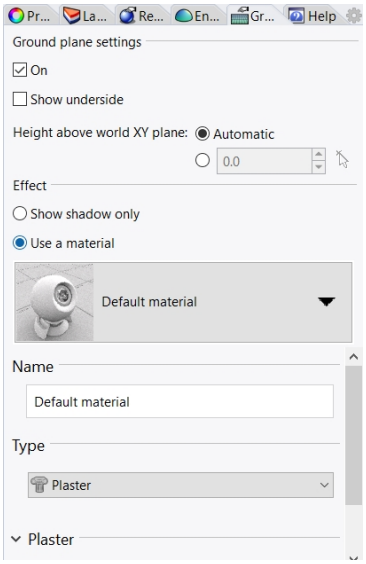
Set up the rendering properties

The rendering properties include environment settings, render, and ambient light settings.

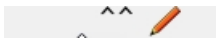
1. Open the model **Mug.3dm**.
2. On the **Panels** menu, click **Environments** and **Ground Plane** to open the panels we will use to set a background environment, and to add an infinite ground plane to the scene.
   This can also be accomplished by right-clicking on the **Properties** panel tab.
3. In the **Environments** panel, set the background to **Environment** by picking on the yellow alert at the top of the Environment panel. It will read:
   The current background image is set to Solid Color. Switch the Background model to Environment.
4. Next click [+] to add an environment and from the pop-up menu, click **Import from Environment Library**.
5. In the **Open** dialog box, double-click **Environments**, scroll though the available Environments.
6. Click **Rhino Interior.renv**, and then click **Open**.
7. To set it current Environment, in the **Environments** panel, double click on the **Rhino Interior**
   It is now the current Environment and the yellow corners mean that **Rhino Interior** is the current environment.



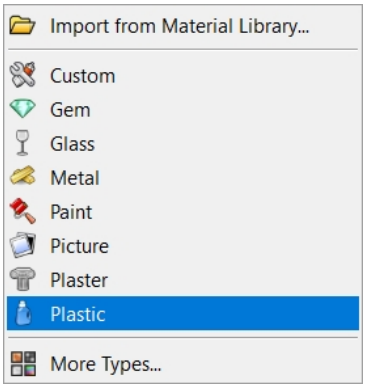
8. In the **Ground Plane** panel, check the **On** option.
   In the **Effect** section, select **Use a material**.
   The ground plane will automatically be set for the **Default material**.



9. Set the **Perspective** viewport to the **Rendered** display mode.

10. From the **Render** menu click **Render**.
11. In the **Environments** panel, double click on the **Studio** to set it to the current Environment.
12. To edit the Environment, click with icon at the bottom of the **Environment** panel. 
13. Under the **Rotation** section, set the angle to **45**.

14. In **Options** dialog, on the **Render** page, scroll down to the **Lighting** section, uncheck the **Skylight** option.
15. On the **Ground Plane** panel, in the **Effect** section enable **Use a Material**.
16. Pick the arrow to the right of **Default Material**, and "**+**".
17. From the pop-up menu, pick the **Plastic** material template.



18. Under the **Plastic** section, set the **Reflectivity** slider to 15%
19. In the **Materials** panel, scroll to **Plastic** and rename material to **Base_white**.
20. From the **Render** menu click **Render**.

Assign materials to layers

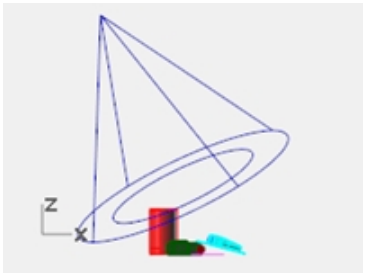
1. In the **Layers** panel, select the **Floss Blister** layer, and click in its **Material** column.
2. In the **Layer Material** dialog box, choose **Thin Clear Plastic** from the drop-down list, and click OK.
3. In the **Layers** panel, select the **Floss Container** and **Toothpaste Tube** layers, and click in the **Material** column of one of them.
4. In the **Layer Material** dialog box, choose **White Shiny** from the drop-down list, and click OK.
5. From the **Render** menu click **Render**.

Scene lighting

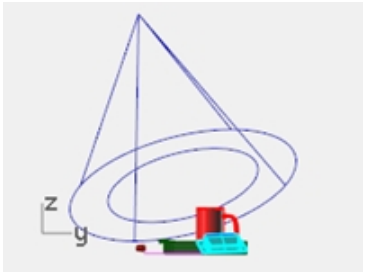
So far, we have used the default lighting in Rhino. This invisible light comes from over the viewer's left shoulder. It is enough to illuminate the model and to give you a starting point. The default light is on only if no other lights are on in the scene and it cannot be modified. In order to control the lighting, we are going to add our own lights.

Add lights

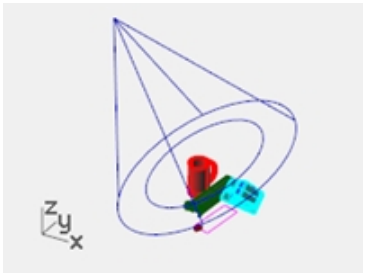
1. On the **Layer** panel, make the **Lights** layer current.
2. On the **Render** menu, click **Create Spotlight**.
3. Make a large spotlight that shines on the scene from the front and slightly above as shown on the right. Use elevator mode, or turn on the spotlight's control points and drag them to move the light into position.



Spotlight, Front view.

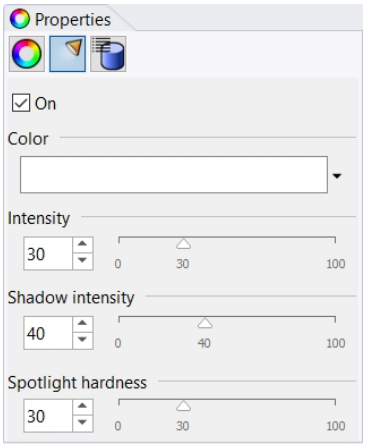


Spotlight, Right view.

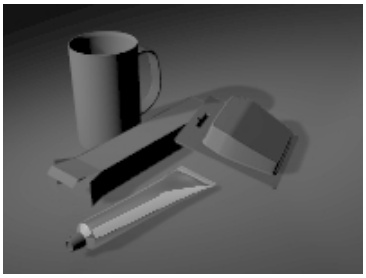


Spotlight ,Perspective view.

4. Adjust the **Properties** of the light as shown:
5. Light intensity=30
   Shadow intensity=40
   Spotlight hardness=30

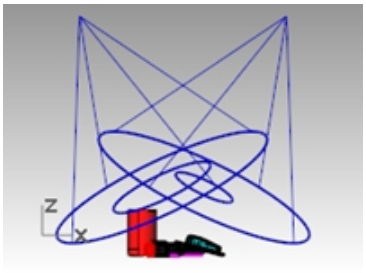


6. **Render** the **Perspective** viewport.
   This makes a nicer image, but two or three lights in a scene improve the rendering. We are going to add another light to create highlights on the mug.



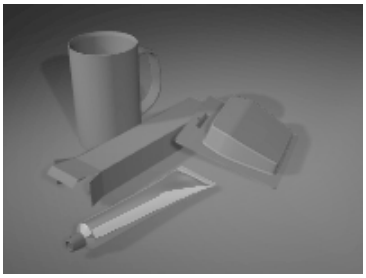
Add a second light

1. Select the first light.
2. In the **Top** viewport, **Mirror** the light across the vertical axis.
3. Adjust the **Properties** of the light as shown:
   Light intensity=20
   Shadow intensity=60
   Spotlight hardness=30



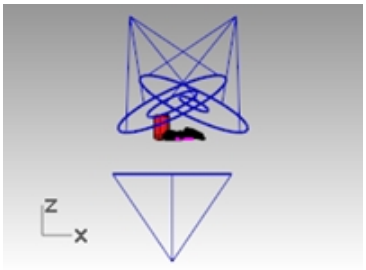
Spotlight, Front view.

4. **Render** the **Perspective** viewport.



Add a third light

1. On the **Render** menu, click **Create Spotlight**.
2. Make a large spotlight that shines on the scene from below.
   This light will be used to add a little light to the underside of the toothpaste tube and the floss packet.
3. Adjust the **Properties** of the light as shown:
   Light intensity=40
   Shadow intensity=0
   Spotlight hardness=30



Spotlight, Front view.

4. **Render** the **Perspective** viewport.
   It is important to turn the shadow intensity to 0 so that the light will penetrate through the ground plane.

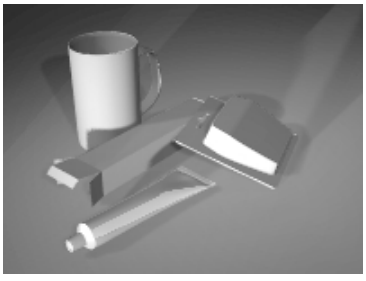


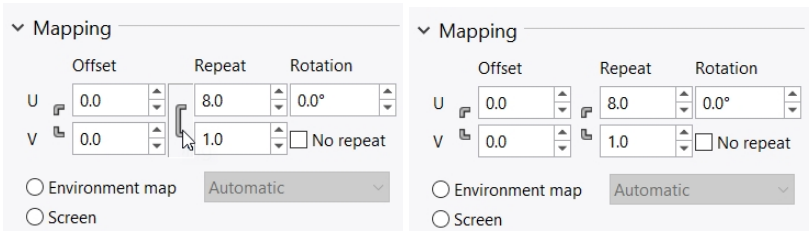
Image and bump maps

Instead of simply using color for your material, you can use an image of a material. You can scan photographs and real objects like wallpaper and carpet, create patterns in a paint program, or use images from libraries of textures from other renderers, or other sources of bitmap images.

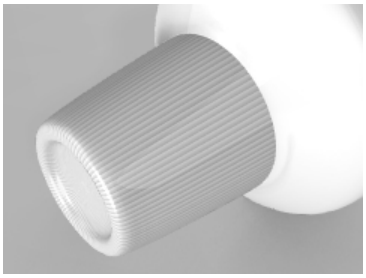
Image mapping uses bitmap images to add detail to the material. You can use images to alter many attributes of the material's surface including its color pattern and apparent three-dimensional surface quality (bump). Procedural bumps add a random roughness or knurled quality to the surface.

Create a new material from an existing material

1. In the **Materials** panel, right-click **White shiny**, and then click **Duplicate**.
2. Name the duplicated material **Toothpaste Cap**.
3. In the **Textures** section, under **Color**, click **(click to assign texture)**.
4. In the **Open** dialog box, double-click on **Tube Bump.png**.
5. In the **Textures** section, click on **Tube Bump** to see additional settings.
6. In the **Mapping** section, set the **U Repeat** to **8**.
   Double-click the lock icon to remove the U-V repeat lock.
   The **V Repeat** should be set to **1**.



7. Assign the new material to the **Toothpaste Cap** layer or assign it to the object. Adjust the mapping as necessary.
8. **Render** the **Perspective** viewport.
   The cap has a grooved appearance. The repeat number determines how close together the grooves appear.



Decals

A decal is the method Rhino uses to apply an image bitmap to a specific area of an object.

The decal mapping type tells Rhino how to project the decal onto your object. The four mapping types, planar, cylindrical, spherical, and UV, are described below.

Decal options

Planar

The planar mapping type is the most common mapping type. It is appropriate when mapping to flat or gently curved objects.

Cylindrical

The cylindrical mapping type is useful for placing decals onto objects that curve in one direction, such as labels on wine bottles.

Spherical

The spherical mapping type is useful for placing images onto objects that curve in two directions. The spherical projection maps the bitmap onto the mapping sphere with the bitmap's vertical axis (height), curving from pole to pole, and the horizontal axis curving around the equator.

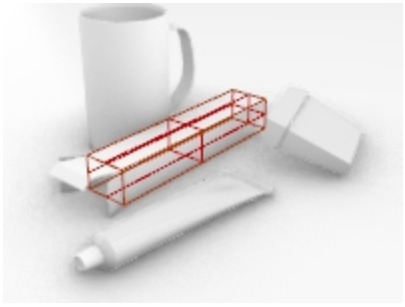
UV

UV mapping stretches the image to fit the whole surface. The U- and V-directions of the surface determine which direction the map is applied. There are no controls.

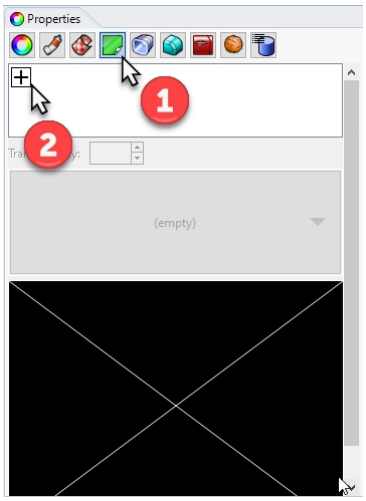
UV mapping works well for organic shapes, hair, skin, and plant structures.

Map a decal with planar projection

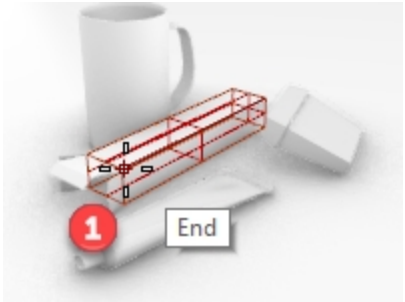
1. Turn on the **Decal reference planes** layer.
2. Select the toothpaste box.



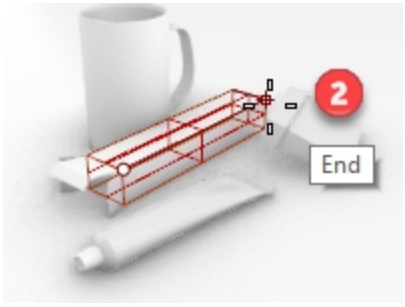
3. In the **Properties** panel, on the **Decals** page click **+** to add a decal.



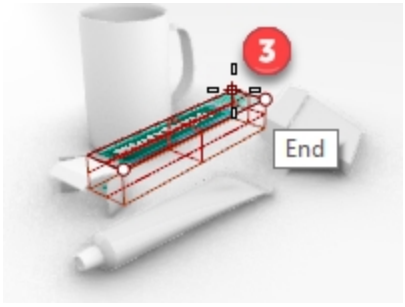
4. Select the **Minty Green-Box Upper.jpg**.
5. Click Open.
6. Click Mapping Style as **Planar**, Direction as **Forward** and click OK.
7. In the command-line, click the **3 point** option.
8. These three points define the decal plane's location and extents. The decal plane must lie on or behind the surface of the object. The decal projects up from the decal plane. Portions of the surface that lie behind the decal plane will not show the decal.
   Using object snaps, pick the location for the decal **(1)**.



9. Pick the **Width** of the decal **(2)**.



10. Pick the **Height** of the decal **(3)**.



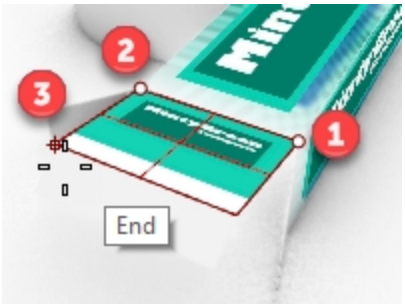
After the decal is placed, you can click the control points on the decal control wireframe to move, rotate, or stretch the decal.

11. Press Enter or right-click to set the location.

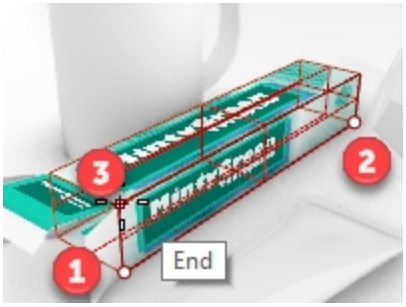
12. Save the model.

Place more decals

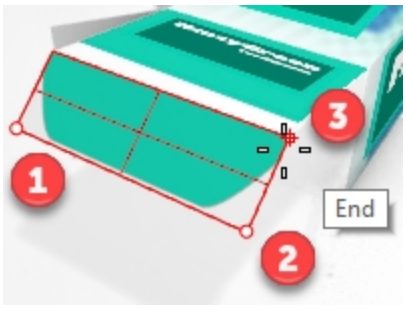
1. Select the surface at the end of the box.
2. In the **Properties** panel, on the **Decals** page click **+** to add a decal.
3. In the command-line, click the **3 point** option to place the **MintyGreen-Box End.png** bitmap at the end of the box.



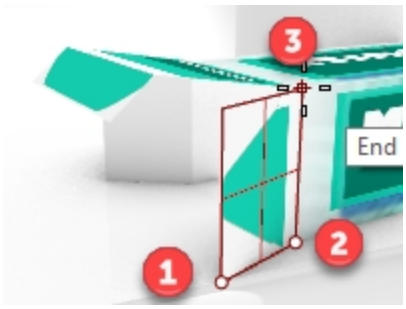
4. With the Decal 3 point option, place the **MintyGreen-Box Side.png** bitmap on the side of the box.

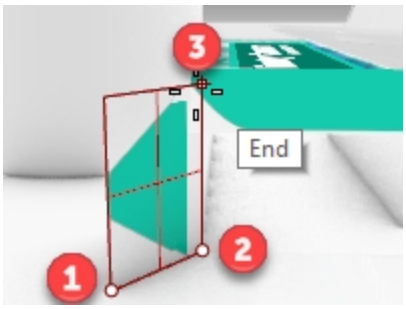


5. With the Decal 3 point option, place the **MintyGreen-TopFlap_RGBA.tif** and for the **Direction** option, select **Both**.



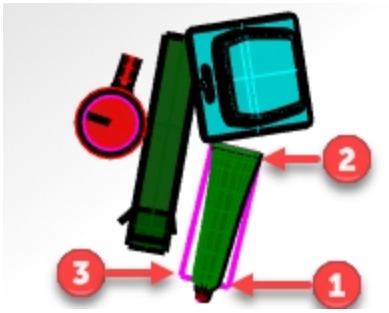
6. With the Decal 3 point option, place the **MintyGreen-SideFlap_RGBA.tif** on each side flap and for the **Direction** option, select **Both**.



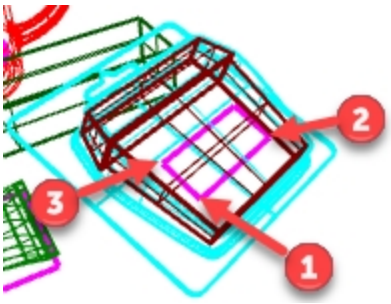


7. Use planar mapping to put the decals on the floss container and the toothpaste tube.
   Turn on the **Decal reference planes** layer.
   The magenta rectangles on the **Decal reference planes** layer were created to assist with placement of the decals **MintyGreen-Floss.png** and **MintyGreen-Tube.png**.

8. Select the tube. In the **Properties** panel, on the **Decals** page click **+** to add a decal.
   Select decal **MintyGreen-Tube.png**.
9. Click Mapping Style as **Planar**, Direction as **Forward** and click OK.
10. In the command-line, click the **3 point** option. Pick the point as follows.



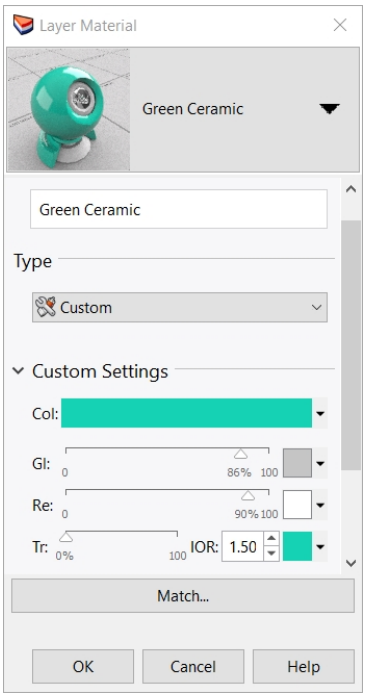
11. Select the box of floss. In the **Properties** panel, on the **Decals** page click **+** to add a decal. Select decal **MintyGreen-Floss.png** .
12. Click Mapping Style as **Planar**, Direction as **Forward** and click OK.
13. In the command-line, click the **3 point** option. Pick the point as follows.



1. **Render** the **Perspective** viewport.

Make a material from scratch and assign it to a layer

1. Open the **Layers** panel.
2. In the **Layers** panel, select the **Mug** layer, and click in the **Material** column.
3. Pick the arrow to the right of **Default Material**, and "**+**".
4. In the **Types** dialog box, click on **Custom** and then click OK.
5. In the **Name** field, type **Green Ceramic**.
6. Set the following:
   Color to Green (R=21, G=210, B=180)
   Gloss color to (R=198, G=247, B=255)
   Reflectivity slider to 30
   Reflectivity color to (R=21, G=225, B=180)
   Gloss finish slider to 86



Map a decal with cylindrical projection

The circle of the mapping cylinder is initially parallel to the current construction plane, and the cylinder's axis is parallel to the construction plane z-axis.

1. Select the **mug**.
2. Select the tube. In the **Properties** panel, on the **Decals** page click **+** to add a decal.
3. Select the **Sailboat-002.tif**.
4. In the **Decal Mapping Style** dialog box, click **Cylindrical**.
5. In the **Perspective** view, use the magenta circle and with the **Center** osnap to click the **Center of cylinder** and the **Radius** or **Diameter** for the decal.
6. Click the start and the end of included angle for the decal placement:
7. Press Enter or right-click to set the location.
8. On the **Decals** page, click the Decal **Sailboat-002.tif** and click **Show Widget**.
9. Use the Widget like you use Gumball to move, scale and rotate the Decal.
10. Click Hide Widget when decal is located where you like.
11. View the result with the **Perspective** viewport set to Render display mode.

The finishing touches

The tooth brush is well organized on to hierarchical layer. You will assign the materials to the layer and geometry on the layer will preview and render in that material.

1. Turn on all the **toothbrush** layers and sublayers. Set the materials to the layer.

Layer	Material
Brush2 Body	Green_ceramic (same as the cup)
Brush2 Bristles Inner	Green_ceramic (same as the cup)

Brush2 Bristles Outer	Porcelain_white
Brush2 Grips	Plastic_white

2. Adjust the materials settings and lighting as needed to get the desired results.
3. **Render** the **Perspective** viewport.

4. Save the rendering to a file.

Chapter 19 - Introduction to Grasshopper

Grasshopper is a visual scripting platform that is included in Rhino 6.

- With Grasshopper, you will write scripts to automate tasks by dragging controls on to a canvas which is its main interface.
- Parameters like the **Number Slider, Graph Mapper, Random** and **Jitter** are used to drive infinite design options.
- The Grasshopper design is immediately previewed in the Rhino's application without generating geometry.
- When final design is selected the geometry is created by "baking" into the Rhino object.

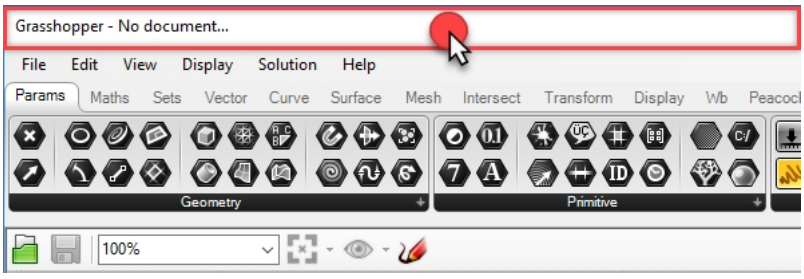
Note: **Bike Wheel.GH** is included in the models folder. You can also print **Bike Wheels.JPG** and follow along with the exercise.

The Bike Wheel

The Grasshopper Canvas

1. Begin a new model with template **Small Objects Inches**.
2. Open the Grasshopper canvas by picking Grasshopper button  in the Standard toolbar or typing: **Grasshopper** in the **Rhino** command-line.

3. Double-click on the top title bar of the Grasshopper window to expand and compress it. However, leave it in the open state. (Windows only feature)

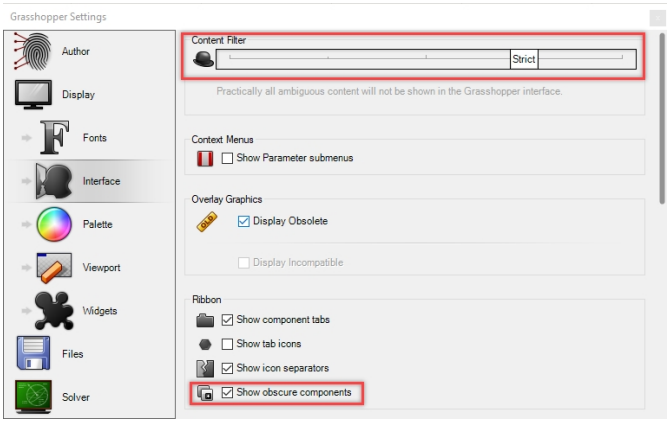


The Grasshopper Settings

There are a couple settings that you will use to control the way the Grasshopper interface will look.

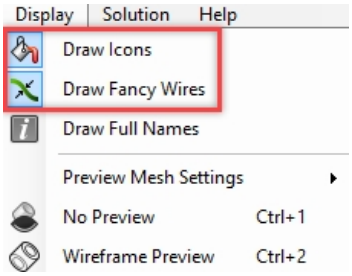
1. From the Grasshopper **File** menu, select **Preferences**.
2. The **Grasshopper Settings** dialog will appear.
3. In the left pane, highlight **Interface**.
4. If you are teaching Rhino to younger students, in the right plane, slide the **Content Filter** to **Strict**. This will display the Grasshopper icons in way that is acceptable to younger users.

5. Check the option **Show obscure components**.



6. Pick the "X' in the upper right corner of the dialog to save and close **Grasshopper Settings**.

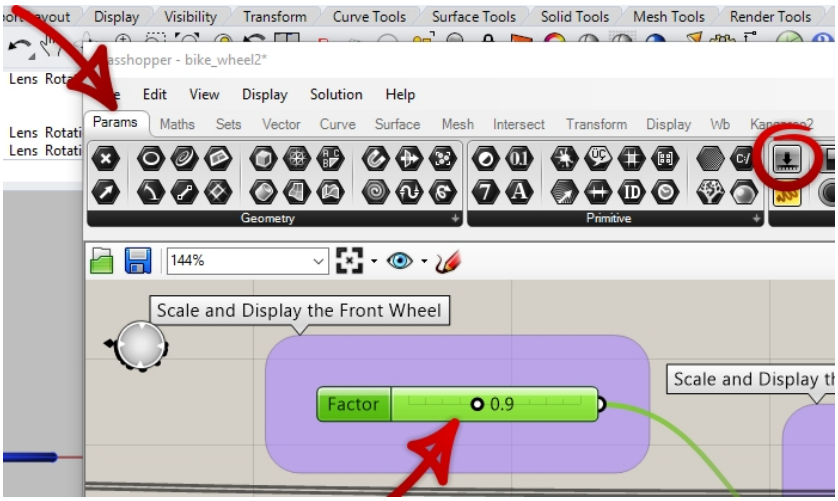
7. On the **Grasshopper** menu, pick **Display**.
8. From the **Display** menu, turn on:
   Draw Icons
   Draw Fancy Wires



The Finder

When you open a completed Grasshopper defintion file (extension .GH) , you can easily trace any component or parameter back to its location on the menu. Grasshopper will display a large red finder arrow on the canvas that will mark where the control is located in the Grasshopper menus.

1. From the Grasshopper **File** menu, select **Open**.
2. Navigate to the files that you downloaded for this training, and open **Bike Wheels.GH**.
3. While hovering over any Grasshopper parameter or component, hold down the Control + Alt keys, while pressing and holding the Left mouse button . The red finder arrows will appear.

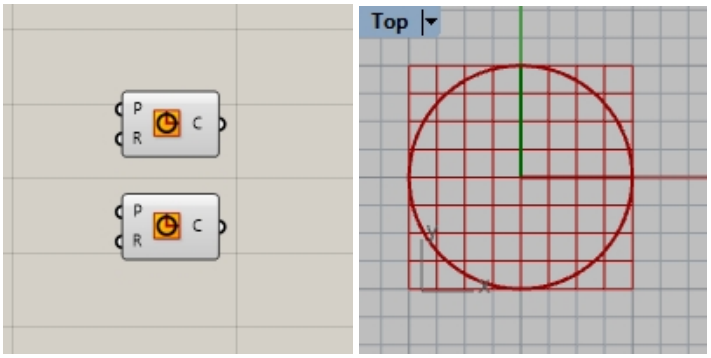


4. You will see the arrows while you have the keys and the mouse button pressed. When you let go, the finder arrow will disappear.
   Note: In Rhino for Mac, use Command + Alt key combination to sctivate the Finder.
   This is a very helpful way to "reverse engineer" a Grasshopper Definition.

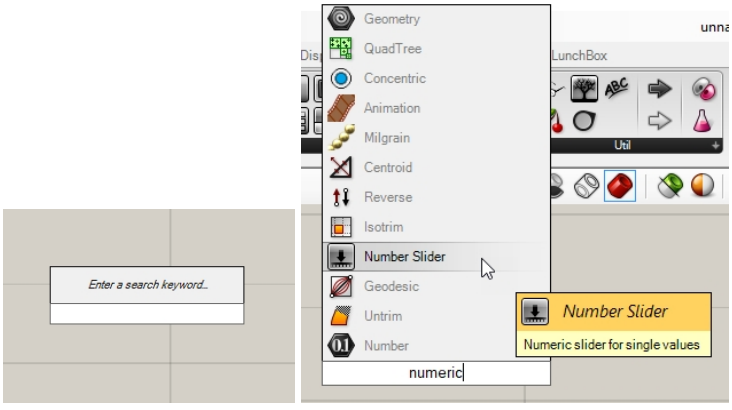
Let's get started with our first simple Grasshopper definition.

Create the Circles

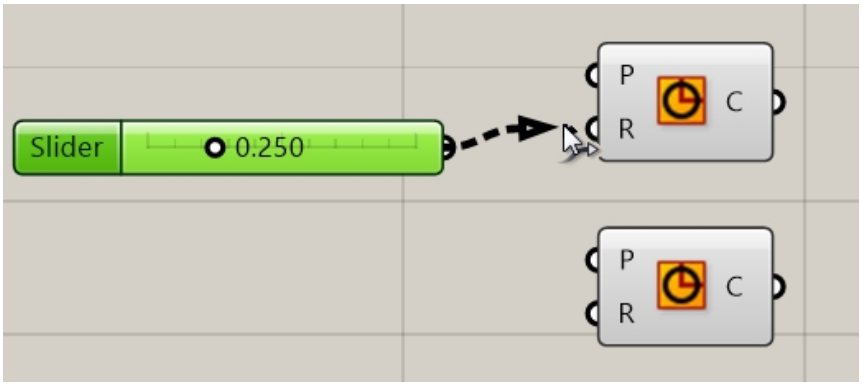
1. On the Grasshopper **File** menu, pick **New Document**.
2. On the Grasshopper **Curve** menu, drag and drop two **Circle** components on to the Grasshopper canvas.



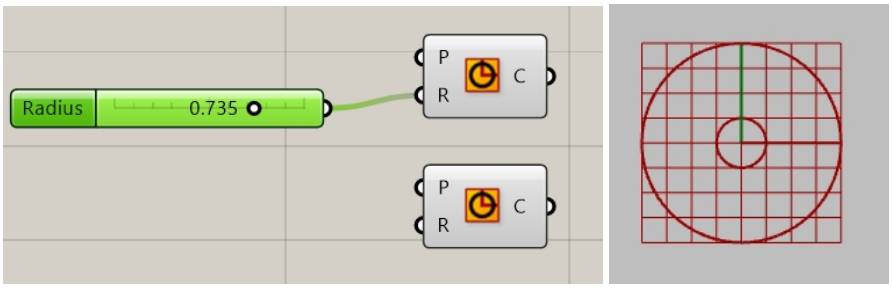
3. Double-click on the Grasshopper canvas to open a dialog box with the prompt **Enter a search keyword.**
4. Type **Number**and pick **Number slider** from the menu.



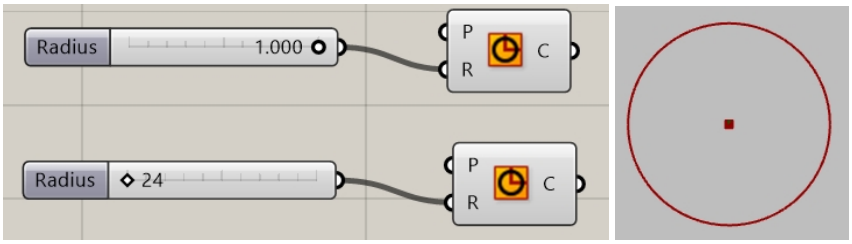
5. A **Number slider** parameter will be added to the canvas.
6. Drag the output connector from the **Number slider** to the input R of the Circle.



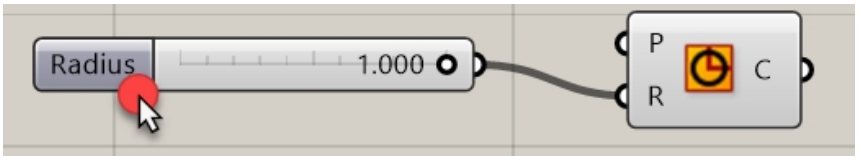
7. Now drag the slider and see the radius of the circle in the **Top** viewport update.



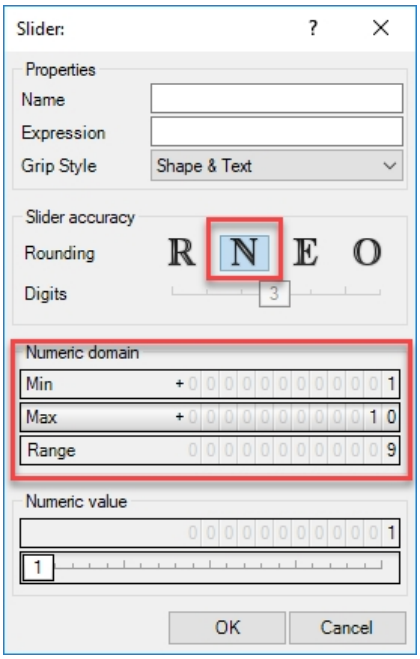
8. To create the second number slider, Double-click on the canvas and type: **24<32<36**.
Plug in the **Number slider** output to the input R for the second circle component.



9. Double-click on the **Radius** label on the first number slider. The Slider dialog will appear.



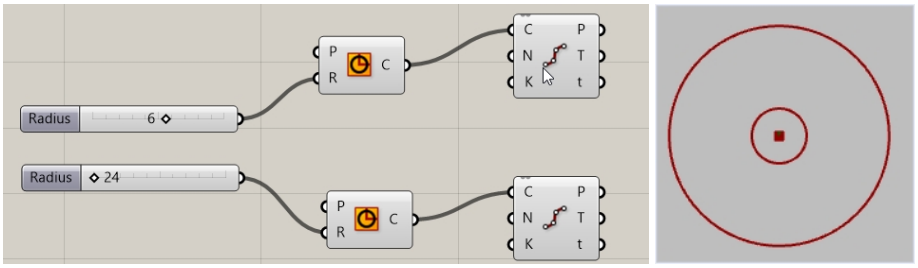
10. Edit the **Min** and **Max** values. Set the **Min** to 1, the **Max** to 10, and the **Rounding** to N, Natural number (Integer).



11. Pick the OK button to close the dialog.
12. Drag the first slider to 6.

Divide the Circle

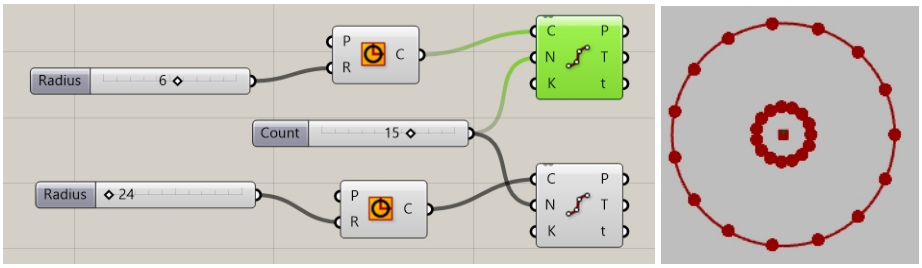
1. On the **Curve** menu, under division, select **Divide Curve** and drop two on to the canvas to the right of the circles. **Hint**: tap the Alt key while dragging a control to copy.)
2. Connect the output circle curve to the **Curve** input on the **Divide Curve** component. Repeat for the second circle.



Connect the Points

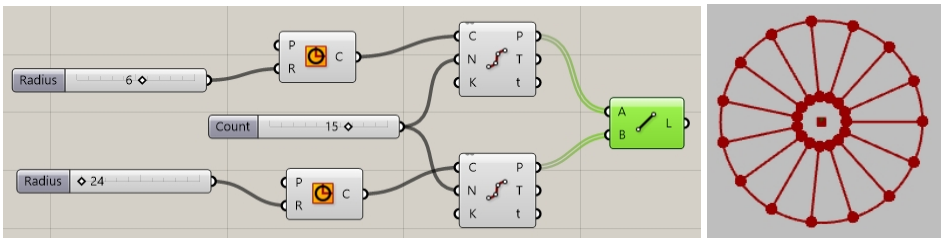
By default, the **Divide** component generates 10 divisions or 10 points on each circle. Now you will make a slider to control that number of points and connect the points to a line component.

1. Double-click on the Grasshopper canvas and create a slider by typing **5<10<20**. This will create a **Number slider** that is set to 10, and whose domain is between 5 & 20.
2. Connect the output of the **Number slider** to the N of each **Divide** component.
3. Now drag the slider and watch the points increase and decrease.

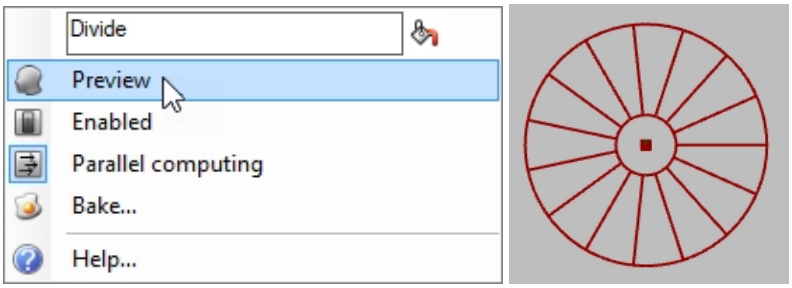


4. On the Grasshopper **Curve** menu, from the **Primitive** Section, select **Line** and drag and drop it to the canvas to the right of the divide component.
5. Connect the output Points from the first divide component to the Line curve input A.

6. Connect the output Points from the second divide component to the Line curve input B.
   Line curves are now connecting the points from both the circle curves



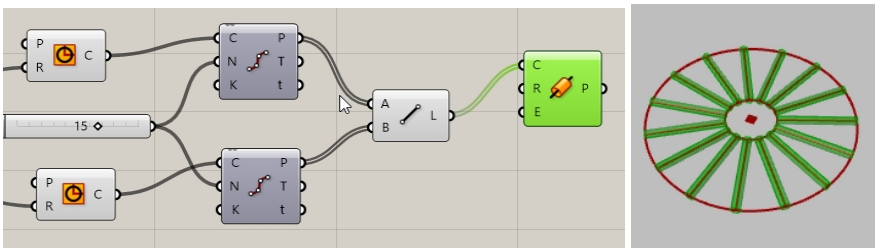
7. Right click on the **Divide** components and pick on **Preview** to disable the preview of points.



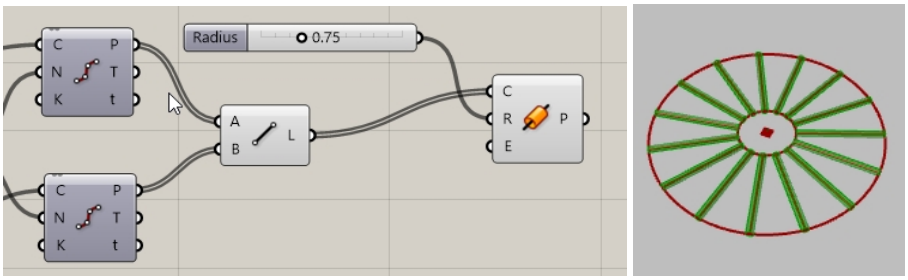
Pipe the Curves

The curves will be used to generate surfaces for the wheel and spokes of the wheel.

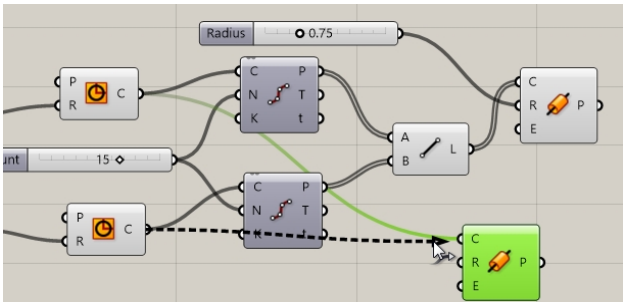
1. On the Grasshopper **Surface** menu, under **Freeform**, select **Pipe** and drag and drop two on to the Grasshopper canvas, to the right of the Line component.
2. Connect the output from Lines to the Curve input of the Pipe component.



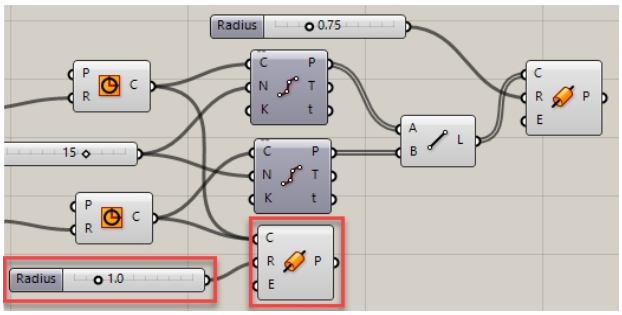
3. Double-click on the Grasshopper canvas and create a slider by typing **.25<1<2.** This creates a **Number slider** that is set to 1, and whose domain is between .25 & 2.00.



4. Connect the output from **Circle** curve to the input **Curve** on the second **Pipe** component.
   Note: You will need to hold down the Shift key to make two connection to an input.



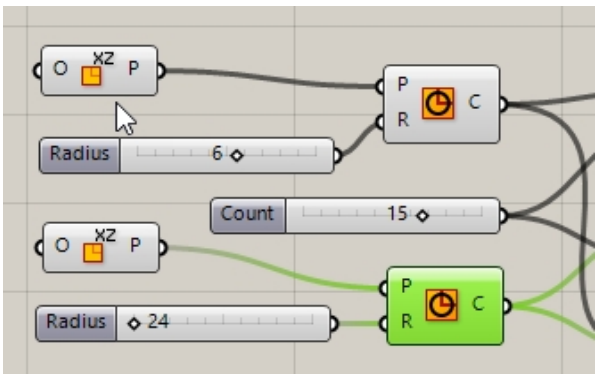
5. Double-click on the Grasshopper canvas and create a slider by typing **.50<1<3**. This creates a **Number slider** that is set to 1, and whose domain is between .50 & 3.00.
6. Connect the output of the last **Number slider** to the input of the radius of the second **Pipe** component.
7. Drag the slider bar and watch the radius of the pipe change.



Orienting the Wheel

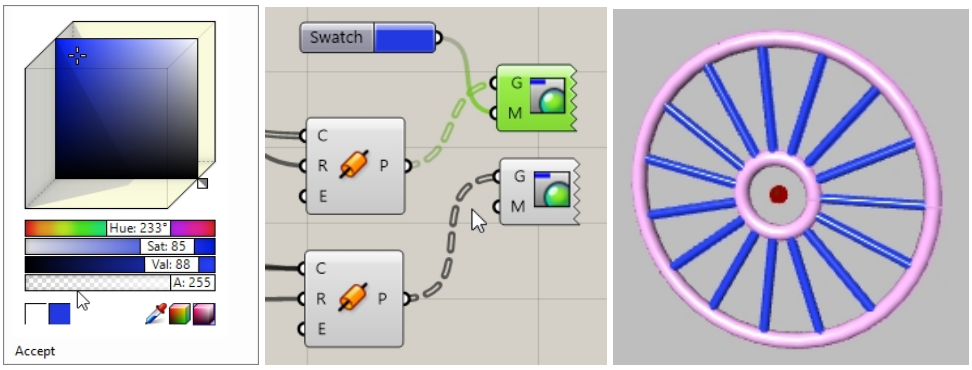
The wheel needs to be oriented parallel to the front or the **XZ CPlane** . To do this, you will go back to the circle and provide a plane to orient the circle.

1. On the Grasshopper **Vector** menu, under **Plane**, select the **XZ CPlane** component and drag two **XZ CPlane** on to the Grasshopper canvas, to the left of the circle component.
2. Connect the **Plane** output from **XZ** plane to the Plane input of the **Circle** component. Repeat for the second circle. The entire wheel design is now oriented in the Front or the **XZ CPlane**.



3. Next, preview the spokes in another color. On the Grasshopper **Display** menu under **Preview**, drag and drop a **Custom Preview** component to the Grasshopper canvas to the right of the spoke pipes.
4. On the Grasshopper **Params** menu under **Input**, drag and drop a **Color Swatch** component to the Grasshopper canvas to the left of the **Custom Preview**. Drag the output from the **Color Swatch** to the input Material override on the **Custom Preview** .
5. Connect the output from **P** from **Pipes** to the input **Geometry** of the **Custom Preview** component.
6. Double-click on the **Color Swatch**

7. Select a custom color from the color picker or drag the sliders for Hue, Saturation, Value, and Alpha transparency. Pick **Accept** when the color preview is as you like.



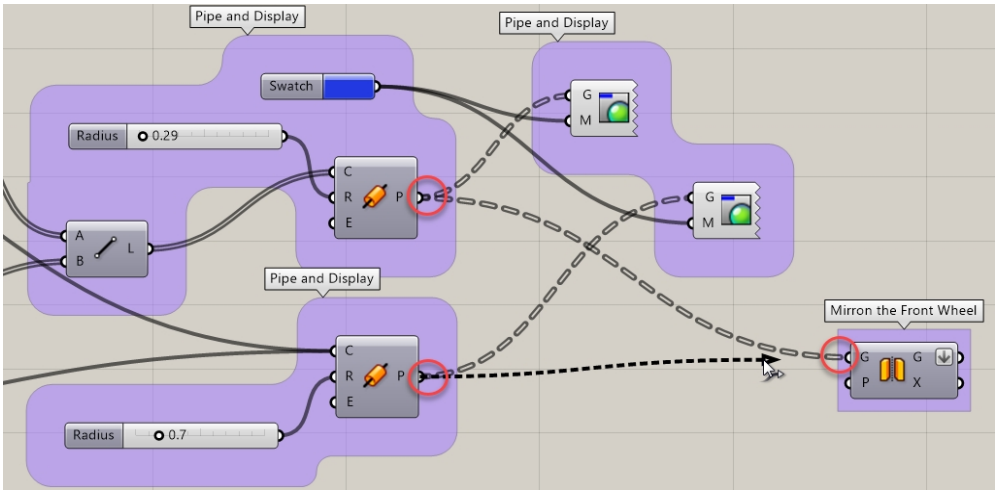
8. Pick **Save** from the Grasshopper **File** menu, or pick the **Save** icon from the Grasshopper canvas toolbar.

9. Save definition as **Wheels.gh**.

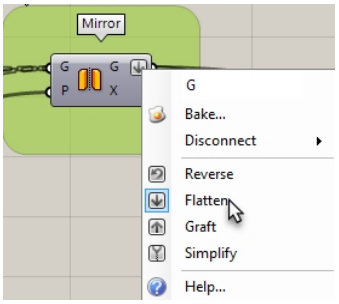
Mirror Front Wheel

The second wheel will be created by mirroring the wheel across a plane that is parallel to the Front or XZ plane. Any changes to the first part of the Grasshopper definition, like wheel size and number of divisions, will iterate through the original and mirrored geometry.

1. On the Grasshopper **Transform** menu, under **Euclidean**, select **Mirror**. Place to the right of the **Pipes**.
2. Connect both the **P** outputs of the **Pipes** component to the **G** input of the **Mirror** component.
   Note: hold down the Shift key to connect multiple inputs to the same connection.

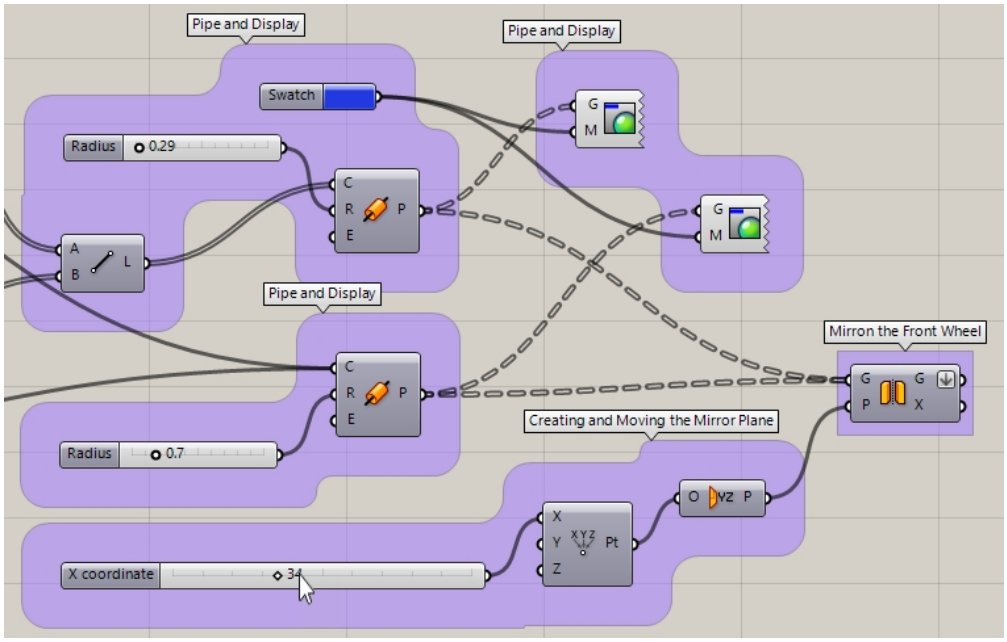


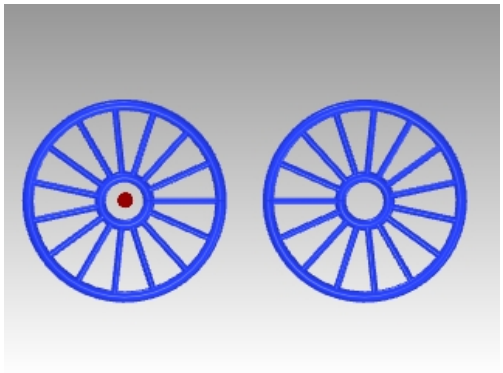
3. Right click over the **G** output of the **Mirror** component, and select **Flatten** from the menu.
   This will convert the two input trees to one list of pipes. The arrow pointing down next to the **G**,indicates that the output has been flattened.



4. On the Grasshopper **Vector** menu, under **Plane**, select **YZ Plane**. Place to the left of the **Mirror**.
5. Connect the output from **YZ plane** to the input of the **Plane** on the **Mirror** component.
6. On the Grasshopper **Vector** menu, under **Point**, select **Construct point**. Place to the left of the **YZ plane**.

7. Double-click on the Grasshopper canvas and create a slider by typing **20<45<60**. This creates a **Number slider** that is set to 45, and whose domain is between 20 and 60.
8. Connect the output from the **Number slider** to the **X coordinate** of the **Construct Point** component. Drag the **X coordinate** slider bar and watch the distance between the original and the mirrored copy update dynamically.

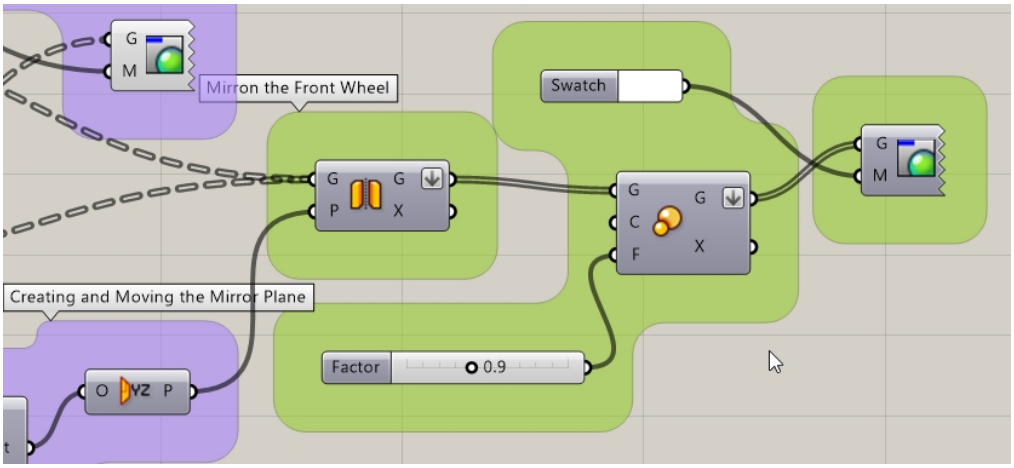




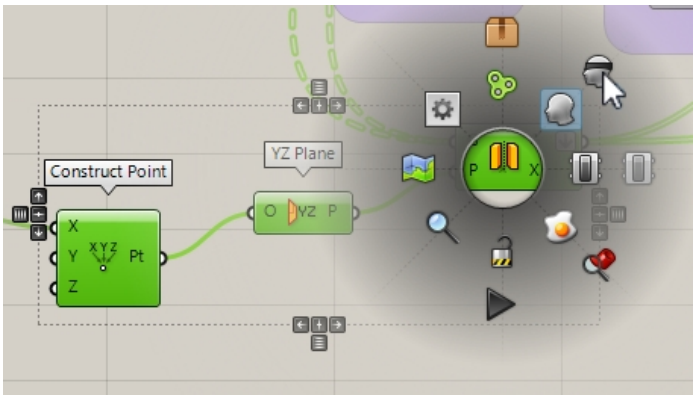
Scale the Front Wheel

The second wheel has been created by mirroring the original wheel across a plane. Next the mirrored wheel will be scaled by a factor making it larger or smaller than the original.

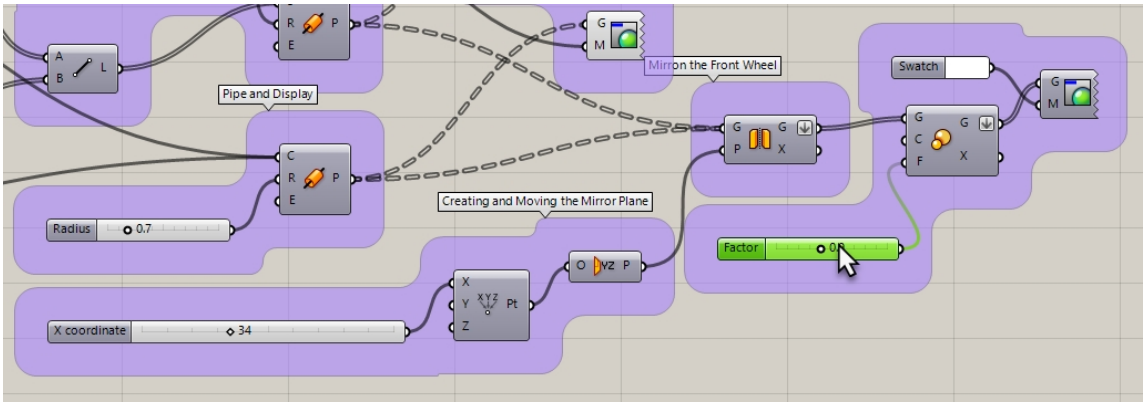
1. On the Grasshopper **Transform** menu, under **Affine**, select **Scale**. Place to the right of the **Mirror**.
2. Connect the **G** output of the **Mirror** component to the **G** input of the **Scale** component.
3. Double-click on the Grasshopper canvas and create a slider by typing **.50<1.00<1.50.**
   This creates a **Number slider** that is set to 1, and whose domain is between .50 and 1.50.
4. Connect the output from the **Number slider** to the Factor input of the **Scale** component.
5. Add a **Custom Preview** component and connect the **Geometry** output from the **Scale** component to the input of the **Custom Preview**.
6. Connect the **Color Swatch** to the **Material** input of the of the **Custom Preview** for the scaled geometry.



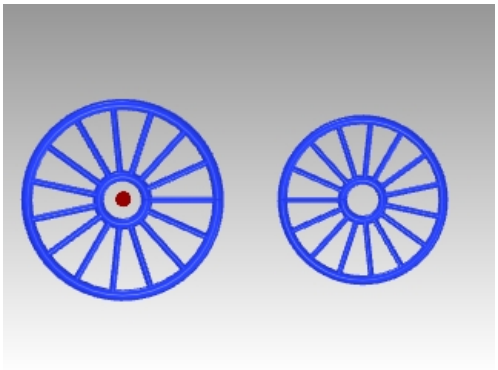
7. Highlight with a crossing window the following components: **Mirror, Point, YZ Plane, Scale,** components.
8. Pick the **middle wheel mouse button** to show the **Radial** menu.
9. Pick the head with the blind fold as shown. This turns the Preview off on all the selected components.



10. Drag the **Factor** slider bar and see the result of the dynamic scale.



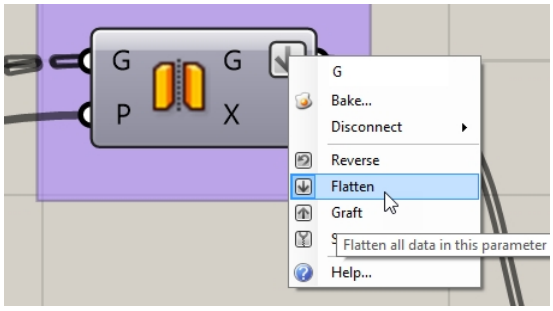
11. The **C** input for Scale is setting the origin for the scale to **0,0,0.** This is not exactly what you were looking for.



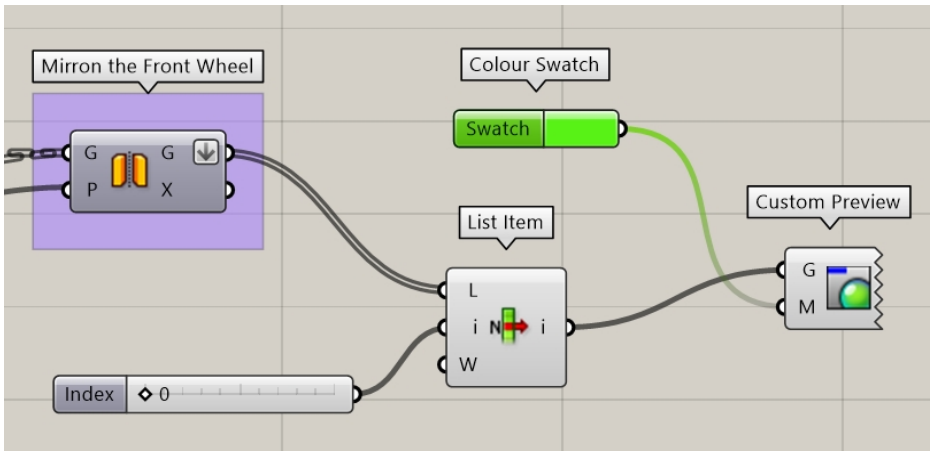
List Item to Select the Tire

You want to scale the front bike wheel and keep the result on the ground. This will take a few more steps. First you will need to locate the outer tire in the "pile" of mirrored geometry.

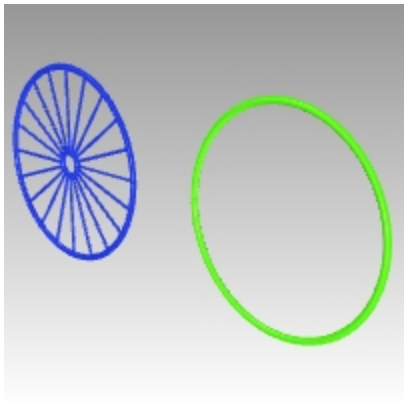
1. On the Grasshopper **Sets** menu, under **List**, select **List Item**. Place to the right of the **Mirror**.
2. Connect the output from **Mirror G** to the input of **List item L**.
3. The output is organized in a Tree structure. **List Item** requires a List. Right-click over the **Mirror G** and select **Flatten** from the menu. This will convert the tree of data to a simple list.



4. Create a slider bar that starts at 0, is set to 0, and ends at the number of divisions plus the two circles. You will use **20** for the number of divisions plus the two circle, and index will go from 0 to 21.
   Double-click on the canvas and type **0<0<21**
5. The slider bar will appear. Connect as input to the input of **List item i**.
6. Add a **Custom Preview** component. Connect the geometry output from the **List Item i** to the input of the **Custom Preview**.
7. Connect the **Color Swatch** to the **Custom Preview M** . Double-click to the **Color Swatch** to select a color to preview.



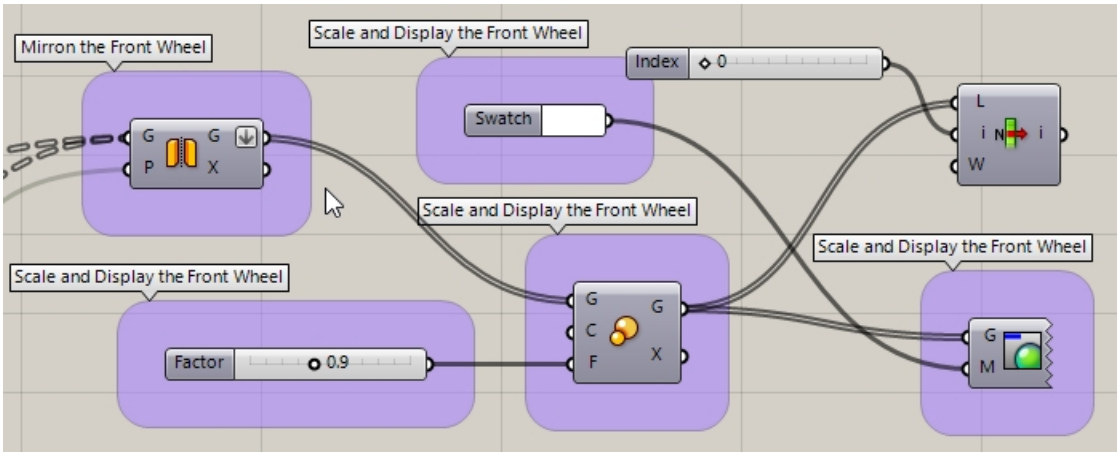
8. Drag the **Number slider** until the outer pipe or wheel's tire is selected.
   (This may vary. In this example, we set the slider to the 0 zero index and the outer circle was selected.)



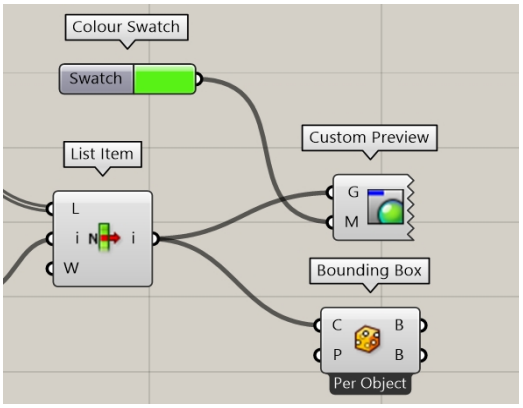
Locating the Bottom Surface of the Wheel Bounding Box

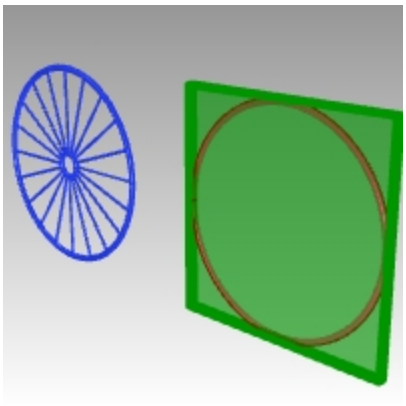
You do not want to scale from the center of the tires. Next, you will use the Bounding Box component to come up with the 3D box that contains the tire. The bottom surface of the bounding box will give Grasshopper the point to be used as the center of the 3D scale.

1. On the **Sets** menu, pick **List Item**. Place the **List Item** component to the right of the **Scale** component on the Grasshopper canvas.

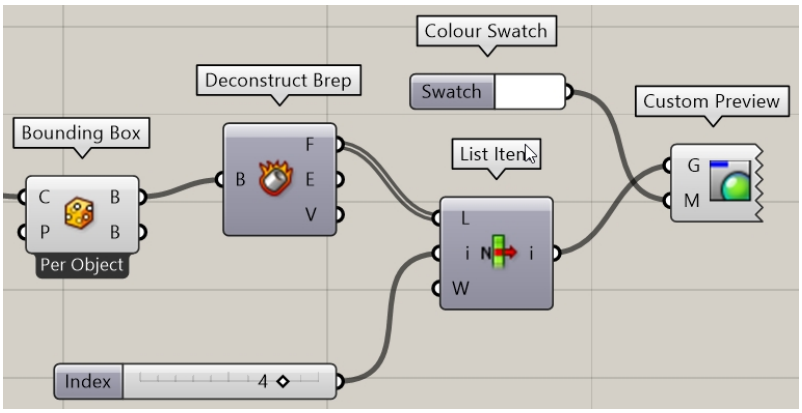


2. Connect the output from **Scale G** to the input **List Item l**
3. Create a **Number Slider** by Double-clicking on the canvas and type **0<0<5**. Connect the slider to the input of **List Item i**.
4. On the Grasshopper **Surface** menu, under **Primitive**, select **Bounding Box**. Place to the right of the **List Item** used above.
5. Connect the output from **List Item i** to the input of **Bounding Box C**.

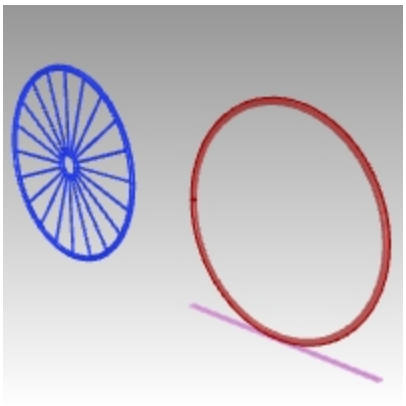




6. On the Grasshopper **Surface** menu, under **Analysis**, select **Deconstruct Brep**. Place to the right of the **Bounding Box**.
7. Connect the output from **Bounding Box B (top)** to the input of **Deconstruct Brep B**.
8. Connect the output from **Deconstruct Brep B** to the input of **List item L**
9. Create a slider bar that starts at 0, is set to 0, and ends 5. This will give us the index of 0 to 5 for the 6 faces of the box.
   Double-click on the canvas and type **0<0<5**
10. The slider bar will appear. Connect as input to the input of **List item i**.
11. Add a **Custom Preview** component and connect the Geometry output from the **List Item i**to the input of the Custom Preview.
12. Connect the **Color Swatch** to the **Custom Preview M** . Double-click to the **Color Swatch** to select a color to preview.



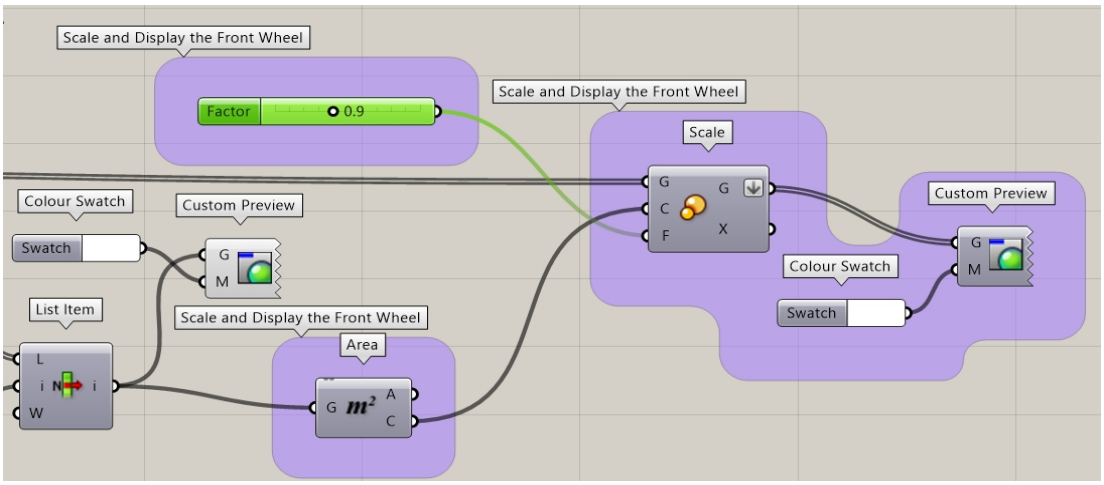
13. Drag the **Number slider** until the bottom surface of the bounding box is selected.



Scaling the Front Bike Wheel from Bottom

You already used the Bounding Box component to come up with the center of the bottom surface of the box. That center will be used for the center of the 3D scale.

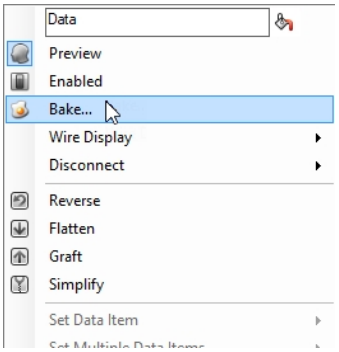
1. On the Grasshopper **Surface** menu, under **Analysis**, select **Area M2**. Place to the right of the **List Item** used above.
2. Connect the output from **List Item i** to the input of **Area m2 G**.
3. Connect the output from **Area m2 C** to the input of **Scale C** or center.
4. Drag the **Number slider** to scale the mirrored wheel from the bottom.



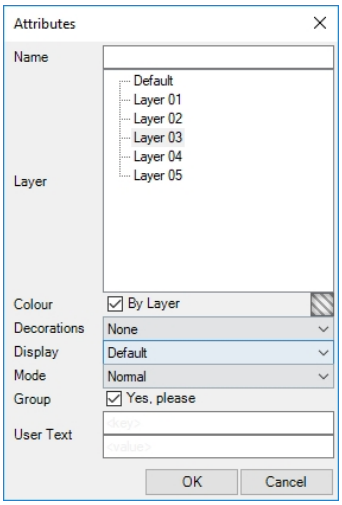
Bake The Wheels

The geometry is still only being previewed in Rhino. To send the geometry to Rhino for editing, rendering, printing and more, you will need to Bake from certain components. You can Bake and select a target layer and group the geometry at the same time.

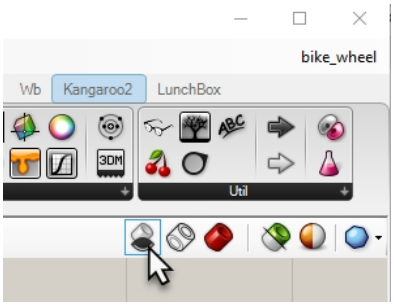
1. On the Grasshopper **Params** menu, under **Primitive**, select **Data**. Place to the right of the **Mirror**.
   The **Data** component will make a copy of the input to be used collectively in another operation, like **Bake**.
2. Connect the output from the both **Pipes** and **Scale** components to the Input of **Data**.
3. Right click over **Data** and select **Bake** from the menu.



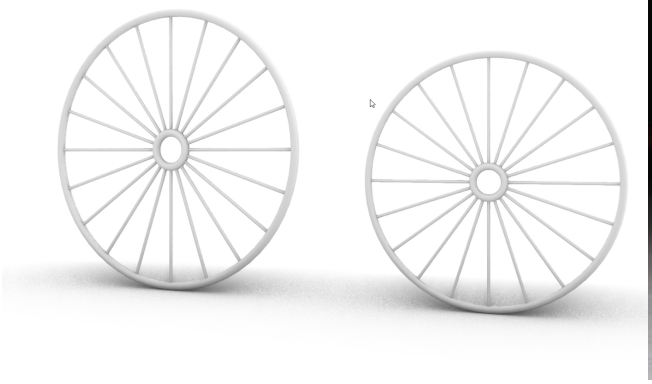
4. Select **Layer 03** and **Yes Please** to **Group** the output.



5. In the upper right corner of the Grasshopper canvas, turn off the preview of the Grasshopper geometry.



6. Double-click on the Grasshopper title bar to compress the canvas.
7. You will now see the model in Rhino.
8. Render the model.



Render the model. Front wheel is scaled in the Grasshopper definition from the bottom of the tire to be smaller than the back wheel.

Note: *design the bike frame and other features of your bike using Rhino.*
See Prof. Steve Jarvis' ART Final Project

Custom bike by Julie Pedalino and ***Pedalino Bicycles****, Lenexa, Kansas.*

https://vimeo.com/172640973