Organic Modeling with T-Splines and Rhino® 4

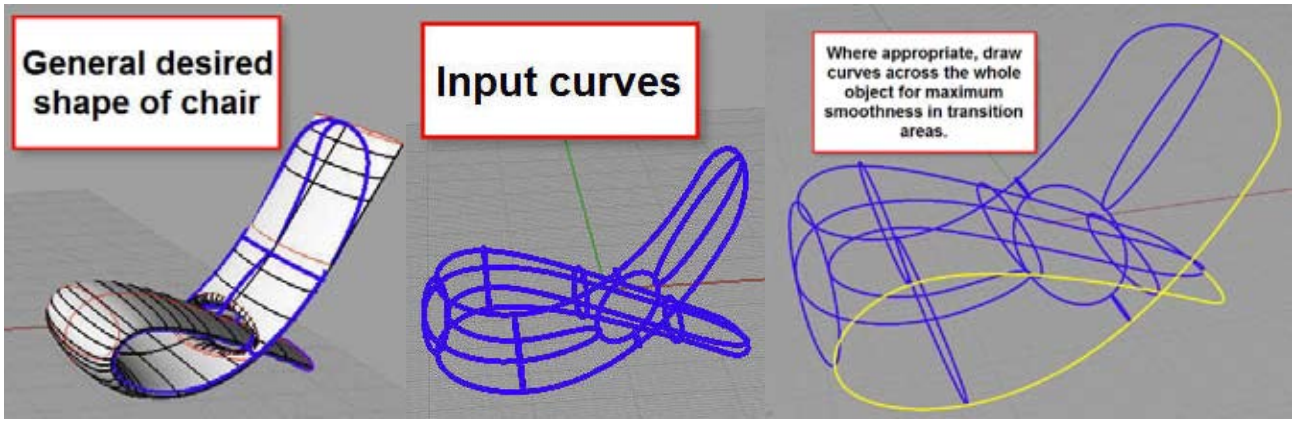
An advanced tutorial in creating a surface from curves

Nicolas Denhez,
Dell
Matt Sederberg,
T-Splines

© Copyright 2007 T-Splines, Inc.

The T-Splines for Rhino plugin brings industrial designers new ways to easily create and modify organic designs. One T-Splines workflow is to create entire, editable surfaces from input curves using the tsSkin command. This tutorial will show you how to use the tsSkin command as you create an avant garde chair desgined by Nicolas Denhez.

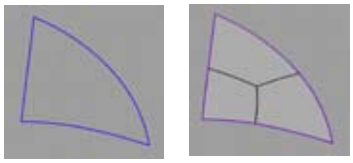
1 Lay out curves to define the surface



First, lay out the curves you will use to define your surface. The following tips may be of assistance.

Lay out curves to define the entire surface
This will yield a smoother surface at intersections than if your curves are only laid out to define sections of the surface. Don't worry about splitting the curves—that will happen in the command. Have intersecting curves intersect as closely as possible. You know the phrase, "Garbage in, Garbage out." If your curves do not do a good job describing the surface, then it will be difficult to make a desirable surface to match the curves.

Create 4-sided faces when possible
Faces that have 3 sides, 5 sides, or more than 5 sides will be broken up automatically with a star point once the model is generated. Star points are not as smooth as other control points, so when possible, use 4-sided faces.



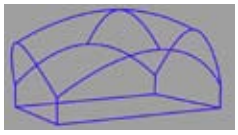
This 3-sided curve section is split into three 4-sided faces upon surface creation

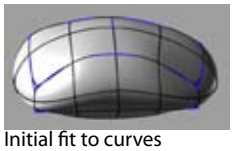
Only one surface may be generated at a time
You can't create two separate surfaces during a single tsSkin session. To create multiple surfaces, just make one surface at once.

When possible, have 4 or 3 curve segments intersect at each intersection
Try not to have more curves intersect at a single point than necessary. Why? Control points that mark an intersection of 4 curve segments, as well as T-Points, where 3 curve segments intersect, are G2 (curvature continuous). Control points that have 5 or more curve segments, as well as those that have 3 curve segments which are star points, are not as smooth. They should only be used when necessary to get the right geometry flow.

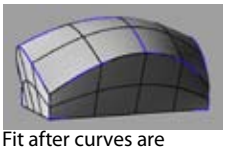
More curves may be necessary
If your first attempt at creating a surface using tsSkin does not yield a good result, you can often increase the quality of your surface simply by increasing the number of curves that define your surface. tsSkin fits a surface to go through curves as closely as possible, but if you have large areas in between those curves, tsSkin can become confused about how the surface should behave there (ie to be rounded or flat). By adding more curves to your object, you can gain additional control about how the surface behaves.

Creased edges must be explicitly creased
The skin command by default will attempt to create a smooth surface--even in areas you would like creased. You must use the "Crease" option inside the command to obtain creases.





Initial fit to curves



Fit after curves are marked as creased

Not valid

The following curve layouts are not allowed in tsSkin:

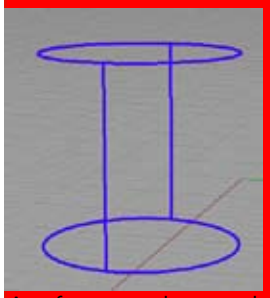
Mobius strips

A mobius strip is the shape tha occurs when you take a long, skinny piece of paper, rotate one end 180 degrees, then tape the two naked edges together. This results in a surface that only has one side (perhaps you've seen these in science museums.) T-Splines does not allow Mobius strips.

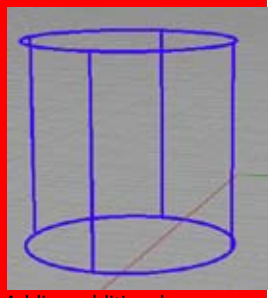
A Mobius strip (courtesy Wikipedia)

Closed surfaces with under four faces

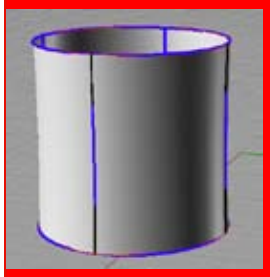
Closed (periodic) surfaces must have at least four faces in the loop.



A surface cannot be created around this cylindrical cuve input because there would just be two faces.



Adding additional curves so there will now be four faces around this cylinder makes this a valid surface.

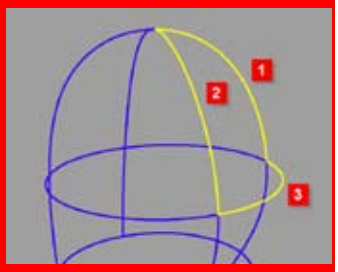


The surface from tsSkin

2-Loops

A surface face comprised of only two curve segments, or loops, is invalid. To fix these situations, add additional curves or split

It is invalid to create a surface face with just two loops.



Adding additional curves will allow a surface to be generated in this area.

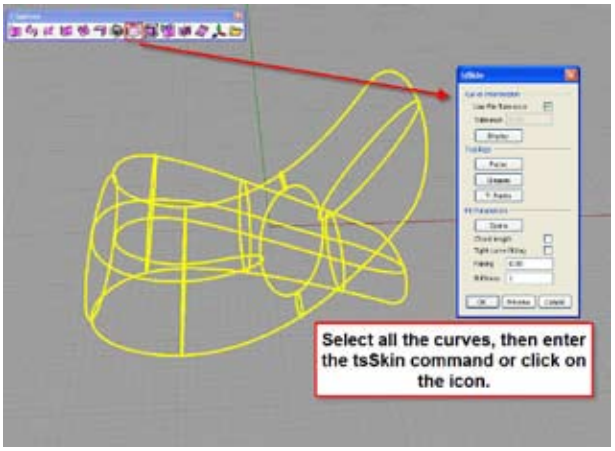
After the curves are laid out, enter the **tsSkin** command. The first thing to do inside the command is to verify the curve intersections.

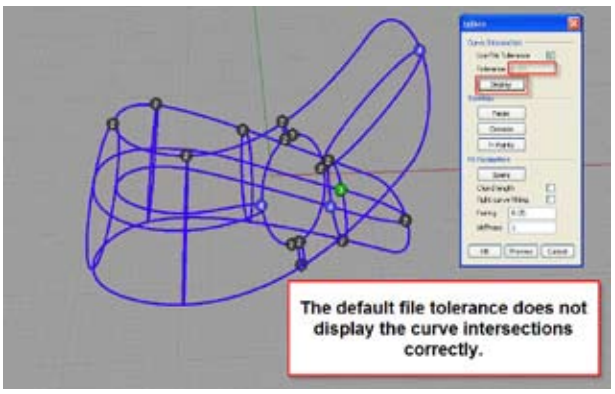
2 Verify the curve intersections

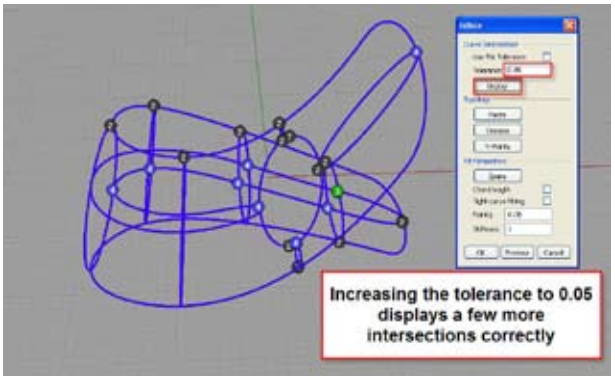
Click on the **Display** button in the Curve Intersection portion of the tsSkin command to see which intersections are being detected. Remember, now is not the time to be passing judgment on whether something is "good" or "bad;" that happened back when you were laying out the curves. At this point, you are just verifying that the computer correctly interprets how the group of curves you gave it intersects.

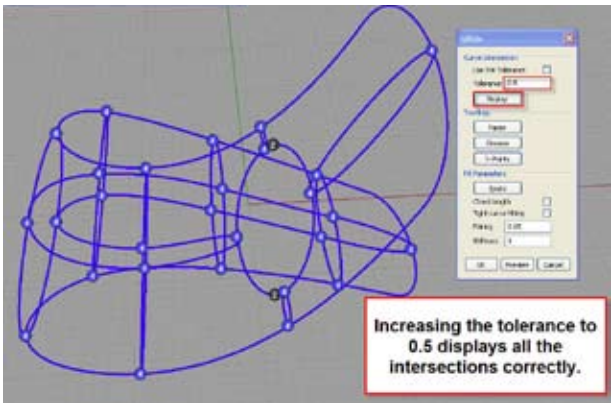
For instance, if you laid out your curves in such a way that four edges should meet at a point, but the computer only recognizes two edges meeting there, you have a problem. The most common problem is that your curves don't actually intersect, they're merely quite close to each other. There are two ways to fix this problem. The first way is to exit the command and reposition your curves until they truly intersect (which may be tedious). A second way is to increase the tolerance used in the tsSkin command. tsSkin uses the Rhino file tolerance by default to determine whether two curves intersect. By unchecking the "Use File Tolerance" box and increasing the number inside the Tolerance box, then you say "even if my curves don't intersect exactly, if they cross within .05 units (or .1, or .5, or whatever) of each other, then treat them as though they intersect."

In this example, when we increase the tolerance to .5, then all the intended curve intersections are finally recognized. It also happens that each intersection has "4" curve segments meeting at it. This is pure happenstance. It is okay to have intersections with "5," "6" or "3" curve segments. Again, these numbers aren't saying whether something is good or bad—the computer is just basically showing its understanding of what you've entered.





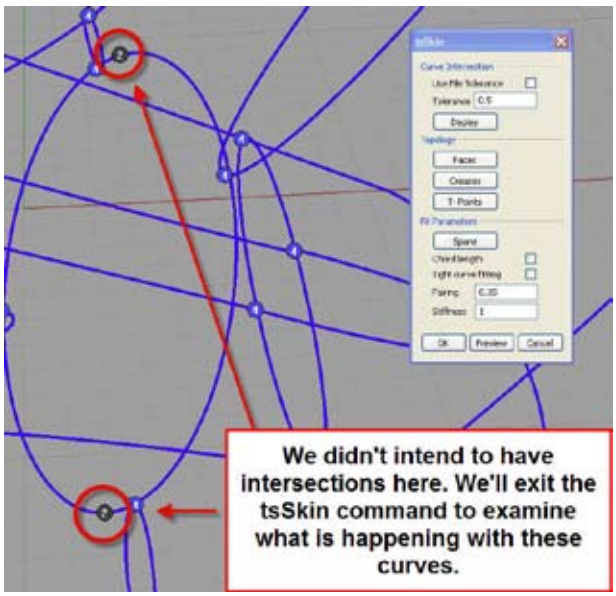


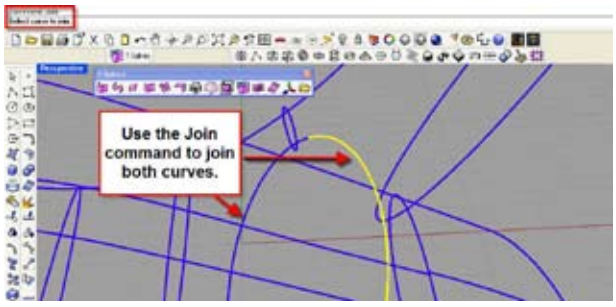


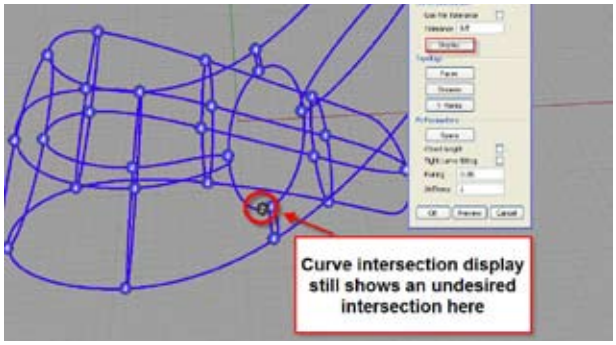
Problem Solving

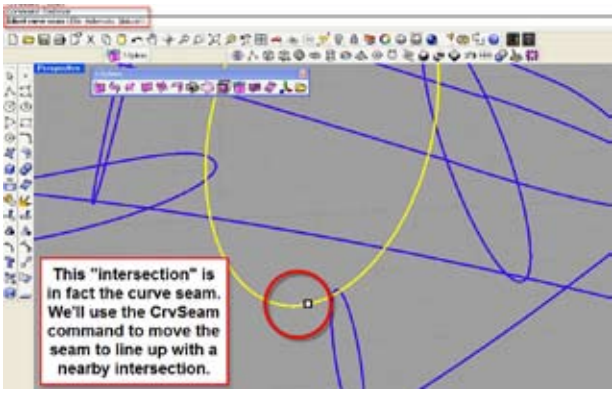
You'll notice that we have two problem areas: we have two "2" dots. The reason that this is a problem is that we didn't intend to have any intersections there. If we exit the tsSkin command and click on the offending curve, we can easily understand why: the curve was somehow split in two. We can **join** the curve and reenter the command.

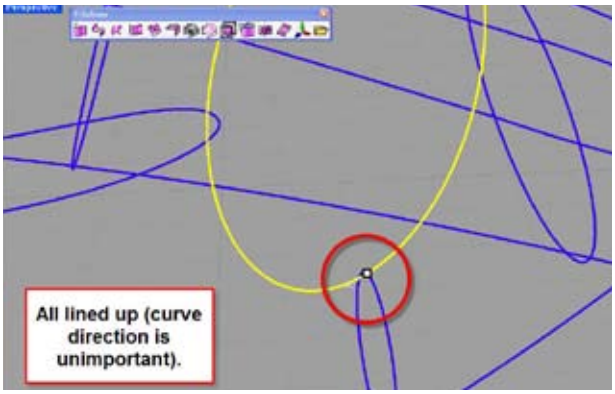
Upon reentering the command, we find that one of those "2" dots is gone but the other is still there. It turns out that this "2" is detecting the seam of the periodic (closed) curve. We need to hide the seam of this curve at a nearby intersection. Entering the Rhino **CrvSeam** command shows us that indeed, the seam of this curve is right where that "2" was. Click and move the seam to the nearest intersection (the direction of the curve doesn't matter).



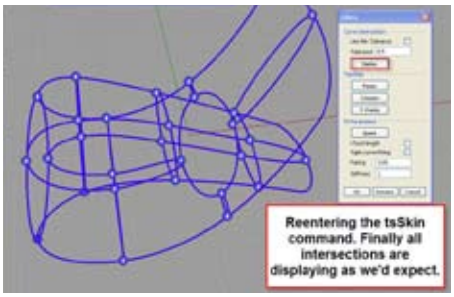








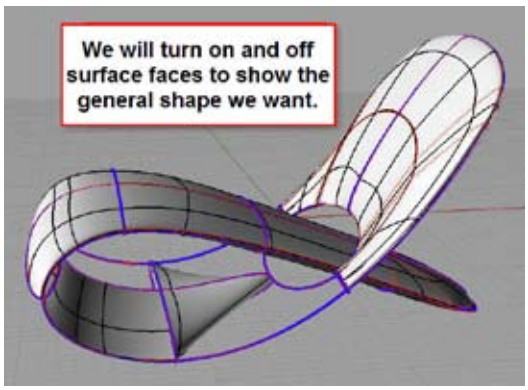
Now, reenter the tsSkin command, and everything finally looks correct when we click on the "Display" button. We're ready to move on.



3 Define topology

In the tsSkin command topology section, we communicate with the computer to determine how the curves we drew should be joined to create a surface. It may not be obvious why this step is necessary, but if someone else stumbled upon your curves, it may be difficult for them to determine exactly how the surface should lie across the curves, as there are many possibilities.. The computer has this problem as well, so all you are doing in this section is applying a rough surface to communicate how the wires should be connected.

Once you click on the "**Faces**" button, you get the computer's best guess on how the curves should be connected to form a surface. This first guess often needs a bit of tweaking.



How does the computer make its guesses? Well, first it looks at the model and tries to identify any "loops" (that is, chains of edges that would encircle a surface) comprised of three curve segments, and fills in surface there. Then it canvasses the model again, looking for any "loops" comprised of four curve segments, and fills in surface inside these loops. That's all it does by default. If you want it to continue filling surface inside of loops, you can change the **Max-AutoFace=** option to have it search for loops with five curve segments, or six curve segments, etc., and fill in surface in these loops automatically.

Often, however, the computer will make some guesses of how the surface should be that you don't agree with. When this is the case, you'll need to turn on or off parts of the surface yourself.

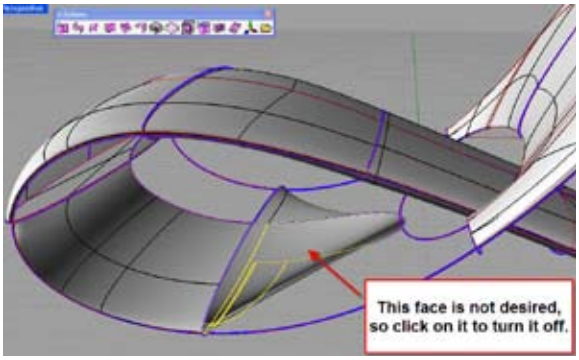
Turning off parts of the surface is easy—just click on the surface and it will disappear. Sometimes the computer will think up a better layout once it realizes you don't want a particular part of the surface, and it will add additional geometry. If you like that, then great! If not, just click on the new surfaces the computer added to turn them off.

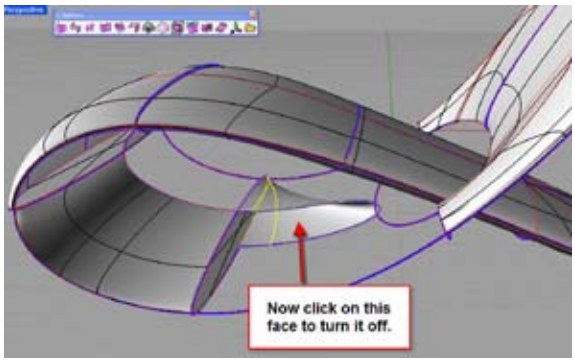
Turning on new parts of the surface is a bit trickier. To let the computer know you'd like to turn on a surface, click on a curve that you would like to be an edge of the new surface (if this curve is an edge of a current part of the surface, holding down "shift" while clicking the curve will ensure that the your click turns "on" a surface instead of turning "off" the adjacent surface by mistake). A list will pop up of all the possible new surface areas that could include the edge you clicked on. As you mouse over the different options in the list, a **loop** highlights on the screen, showing you the boundary of the new part of the surface. This is what is meant by a 4-loop or a 6-loop—it is merely the outer perimeter, or "loop" of the surface, and the number indicates how many curve segments are included.

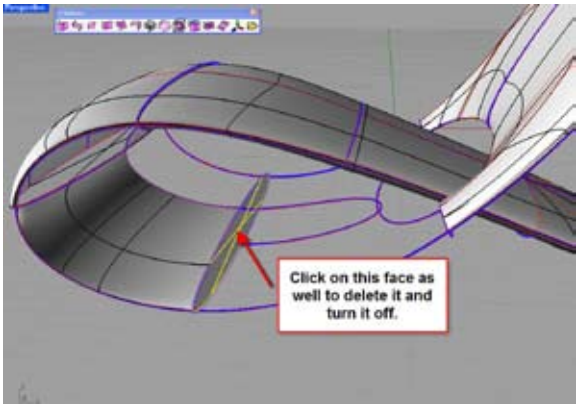
If this list is uncomfortably long, then you can trim it down by decreasing the value of **MaxManualFace**. A value of 5 will suffice here.

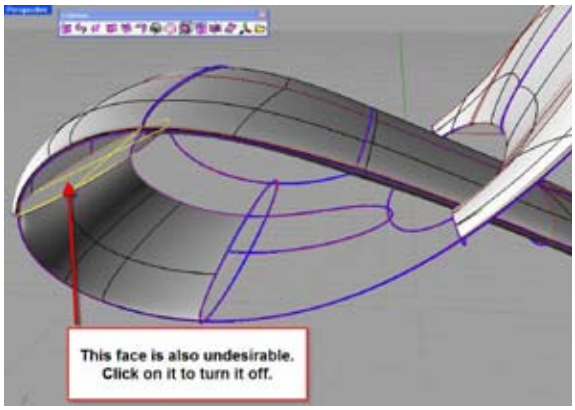
The faces here are just very rough representations of the surface so you know that the computer is connecting the curves correctly. They have no correlation with how smooth the surface will actually be—they're just a fast preview.

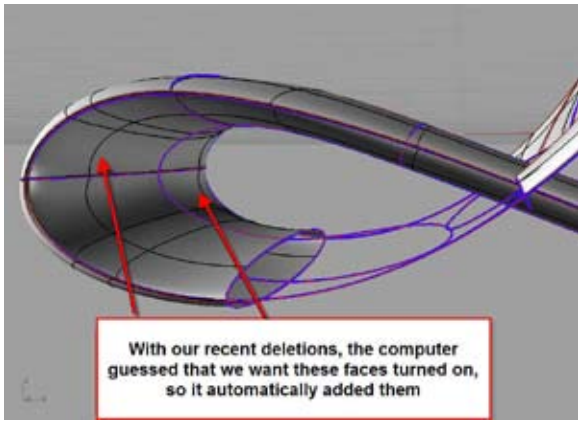
The following pages will show in detail how to turn on and off the surface to define the toplogy.

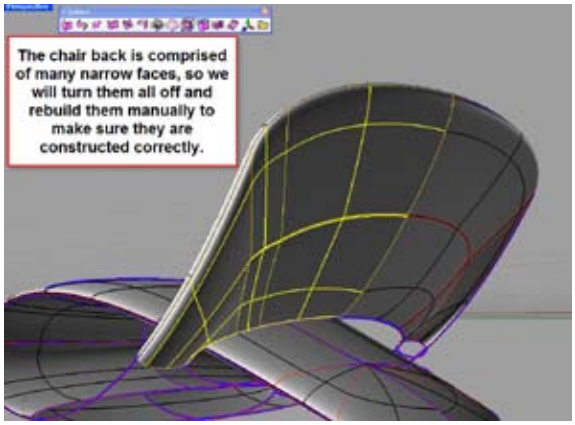




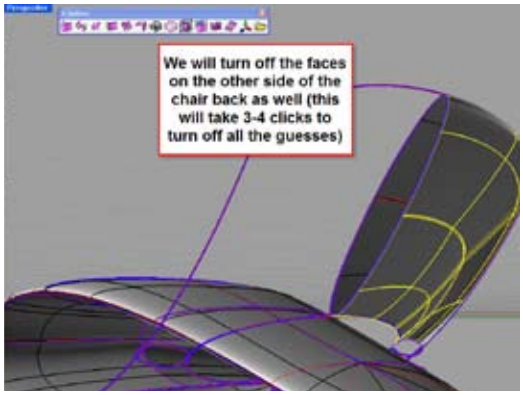


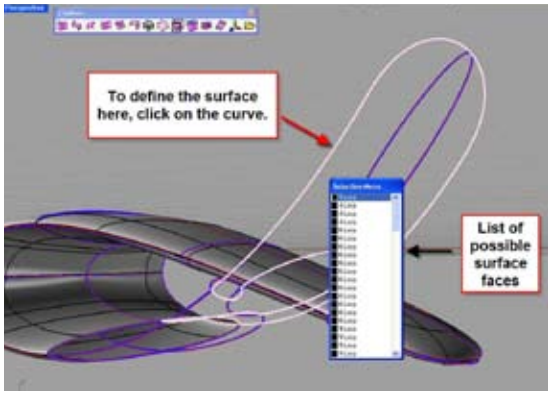


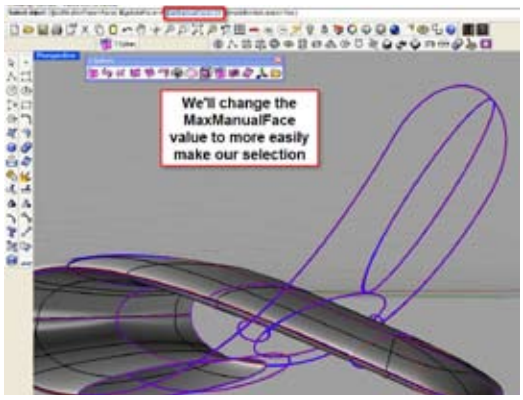


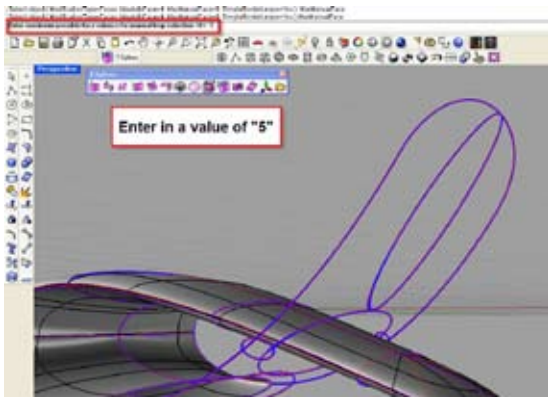


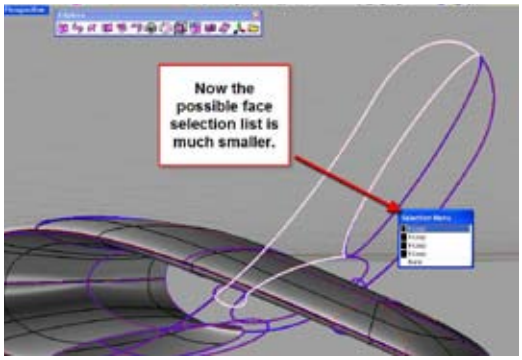
Define topology, continued...

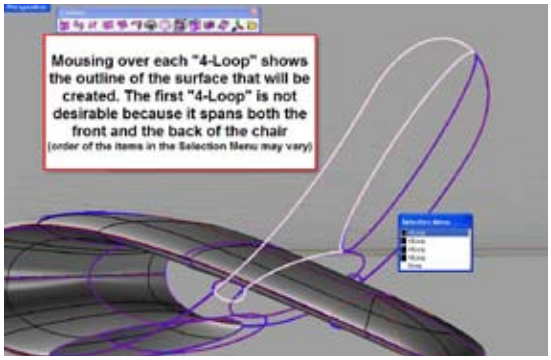


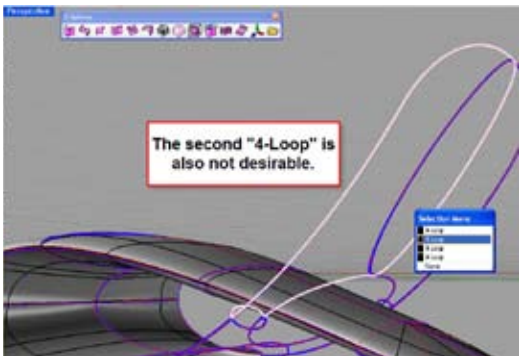


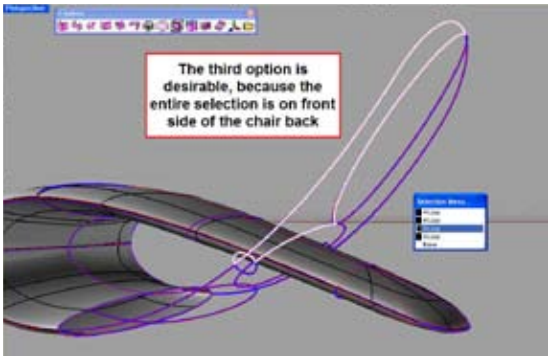




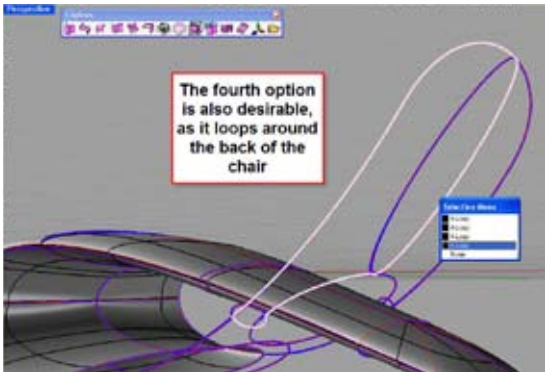


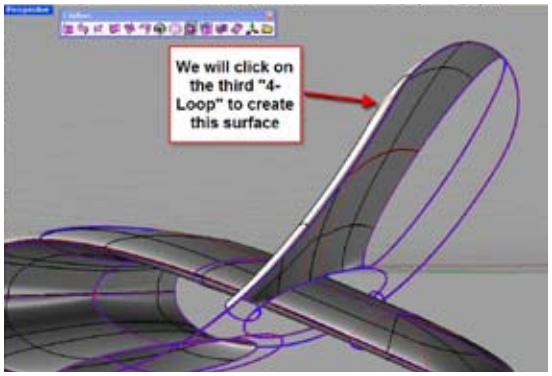


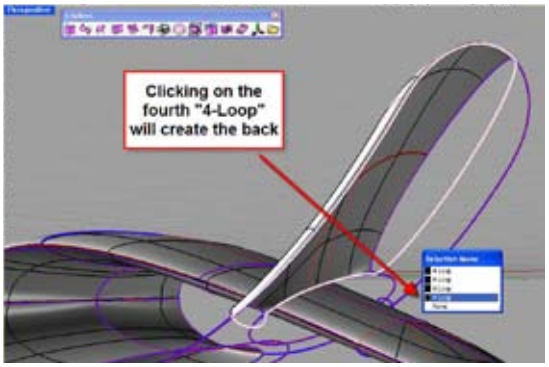


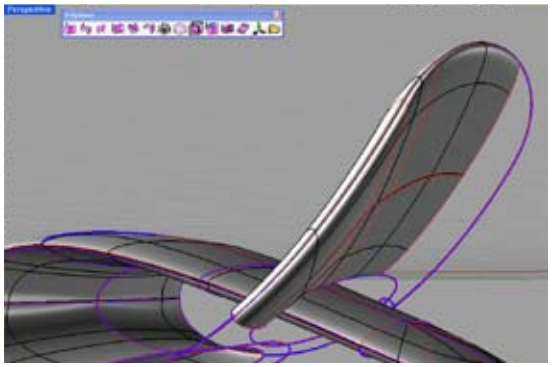


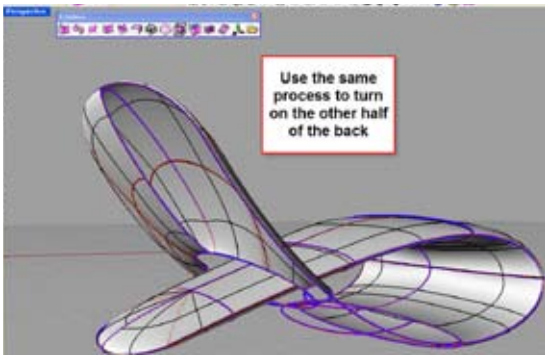
Define topology, continued...

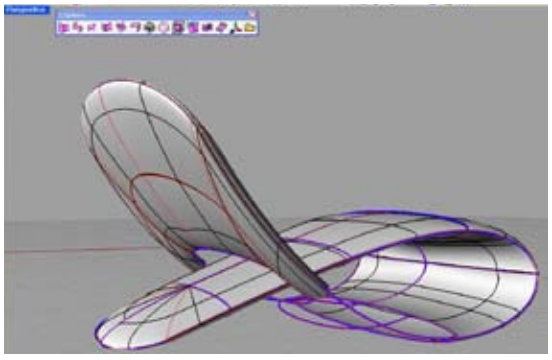


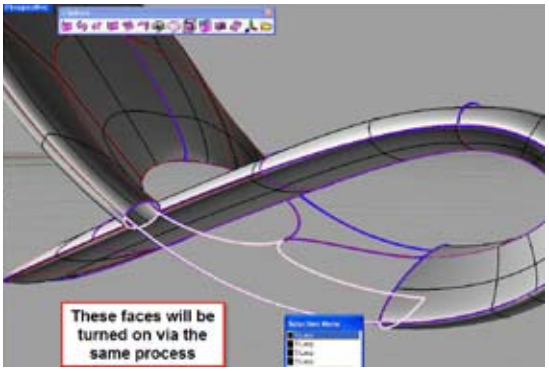


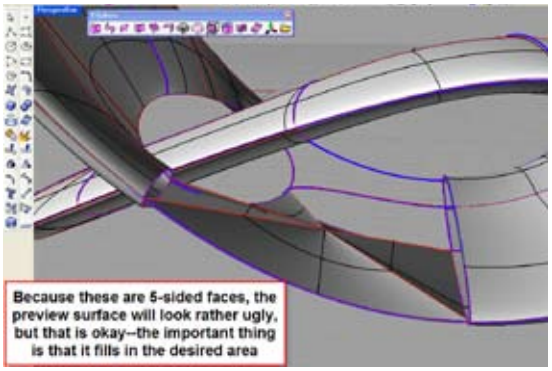




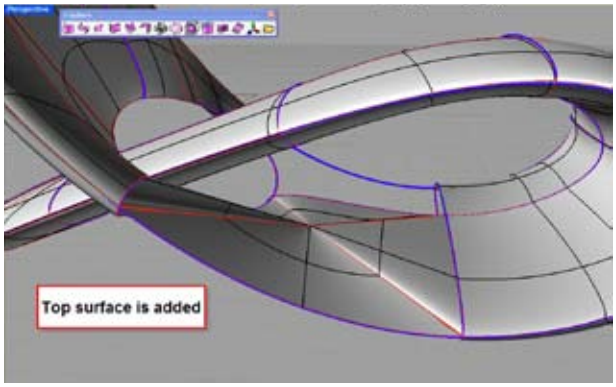


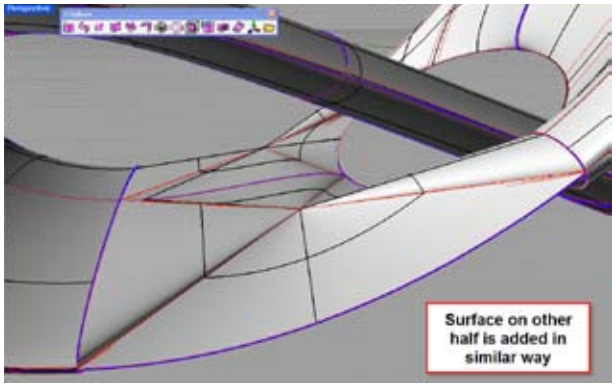


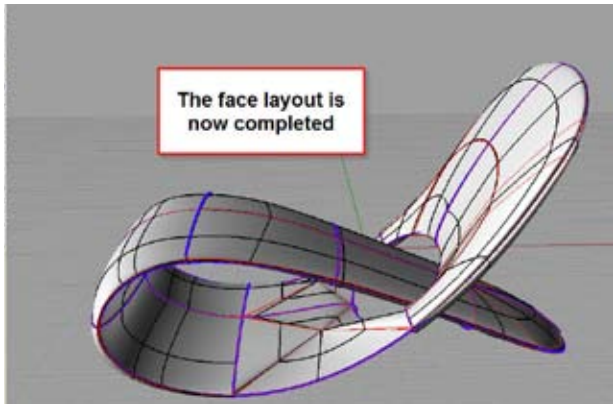




Define topology, continued...



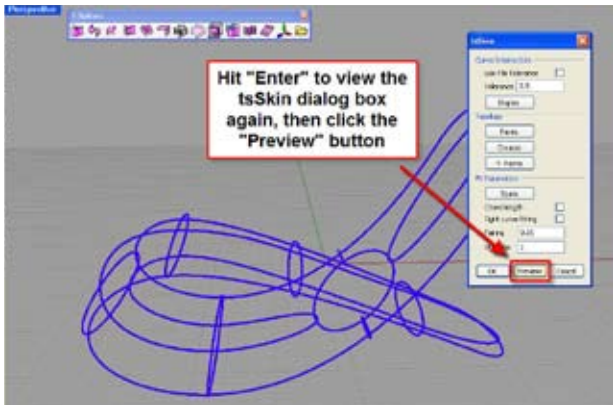


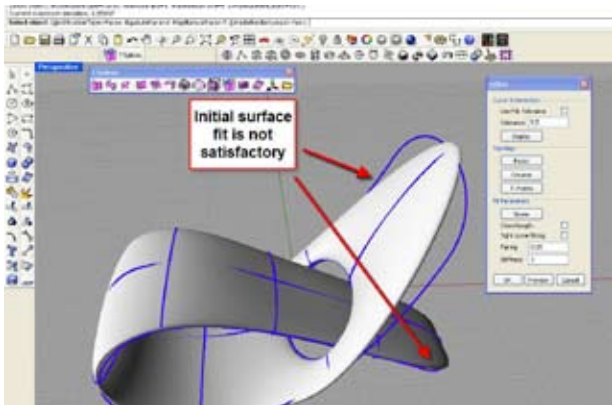


Once you have all the faces of the surface lined up as you would like, you can decide if you want to designate some of the lines as being creased or whether points with three curve segments coming out of them should be T-Points or star points. But since options aren't applicable here, you can just go ahead and preview your surface to see what it looks like.

4 Preview

The initial surface preview is not satisfactory—it doesn't follow the curve inputs very well. The tightness with which the surface matches the curves can be improved by using the **Spans** option.

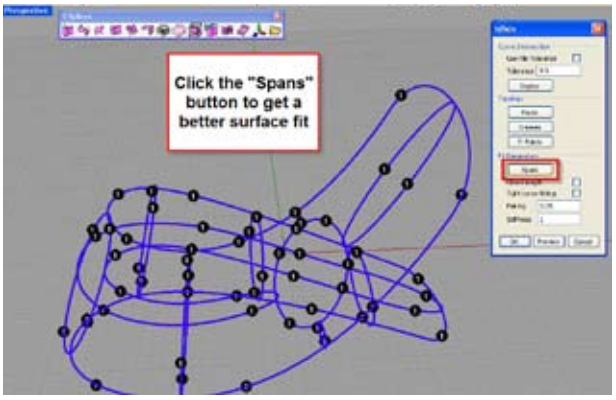


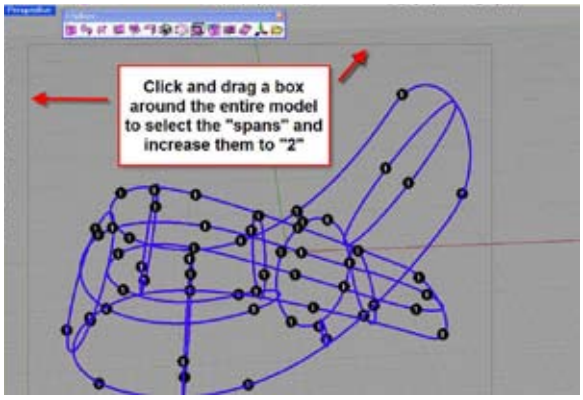


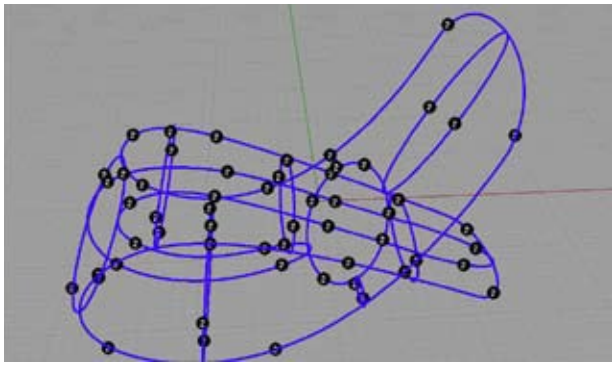
5 Spans

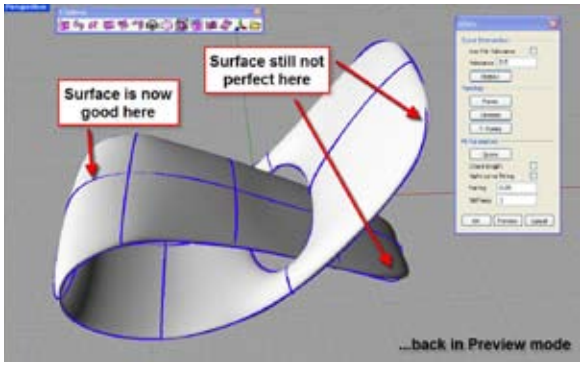
Start by making sure **SpanMode**=Increase, then by increasing each span, which all currently say "1," to "2." This can happen by either clicking on each dot, or by doing a marquee selection and drawing a box around all dots to select them all at once.

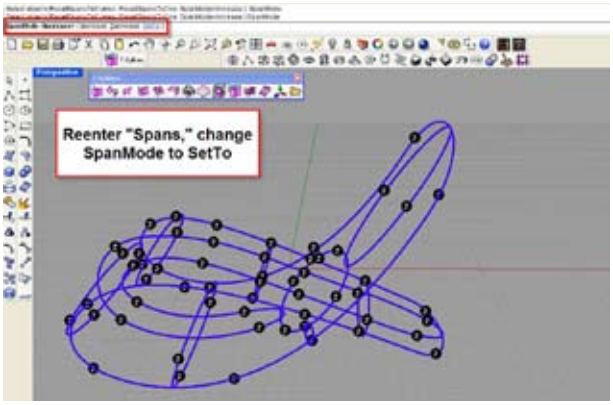
What do "Spans" mean? The spans on each curve indicate how many control points will be on that curve once the surface is generated. If there is just one control point to define a long, bendy curve, most of the curvature will be lost, while adding more control points allows for more curvature. Having "2" control points everywhere produces a much better fit in this example. In fact, the fit for part of the surface is already satisfactory. Clicking on Spans again, we'll now focus on the two ends of the surface where the surface doesn't quite match the curves yet. We can increase the spans here until the surface becomes satisfactory (4-6 spans).

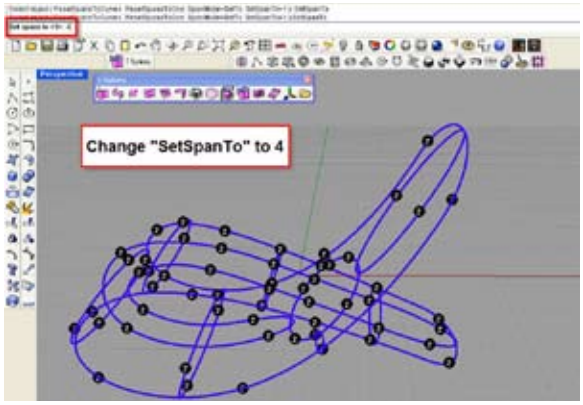


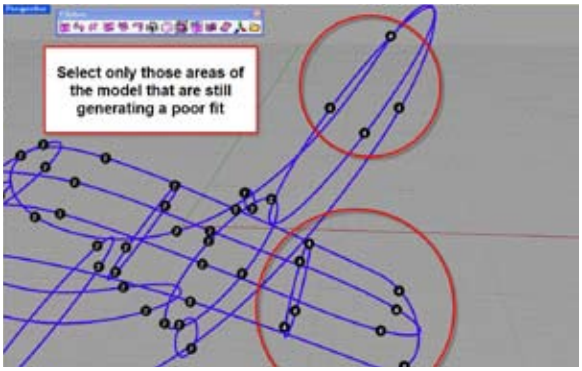


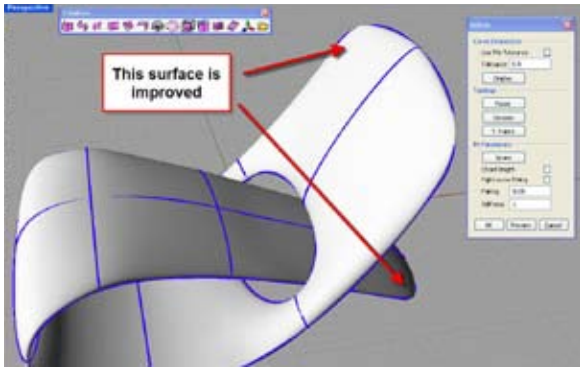




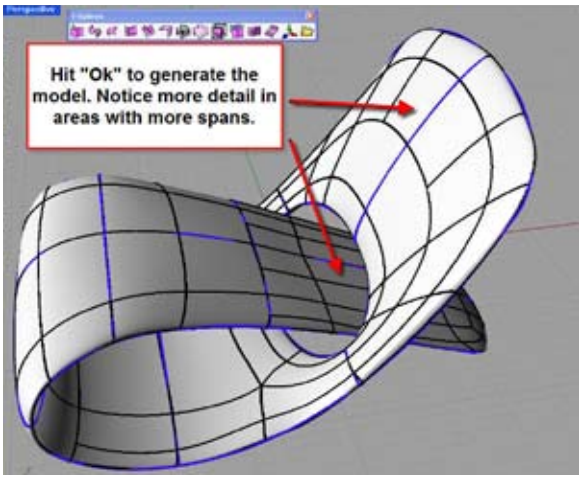


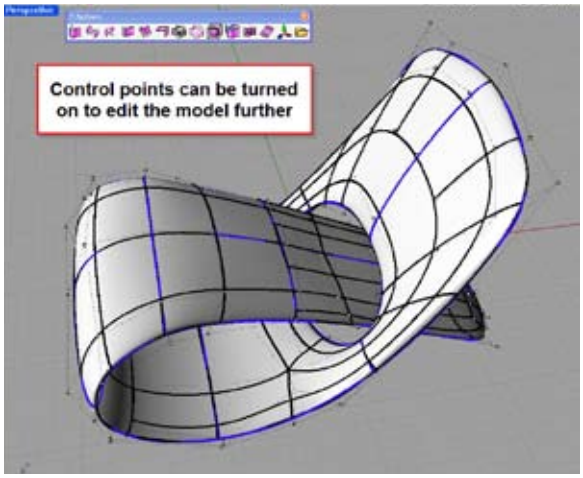






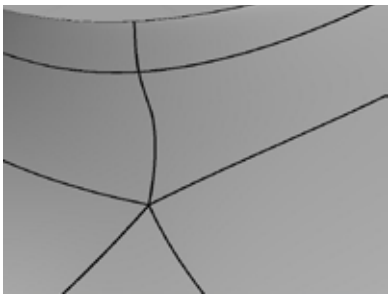
Once the preview surface is satisfactory, the T-Splines surface can be generated by hitting OK. Now, the surface can be further edited by moving control points. Note that there are more control points in areas where there are more spans.





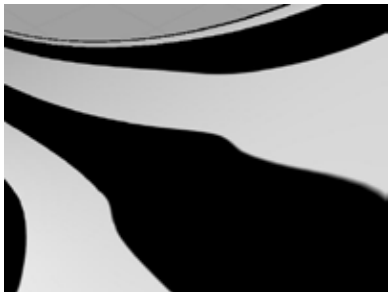
6 Smooth star points

As noted in the "Tips" section earlier, star points are not as smooth as other points in the model. They are G0 smooth by default, but can approach G1 smoothness. To make star points more smooth, use the tsSetStarSmoothness command. A higher number will make the surface smoother, but can also decrease performance time. Also, a higher number will increase the density of the NURBS surfaces created when the T-Spline is converted to a NURBS.



Closeup of star point region of the surface

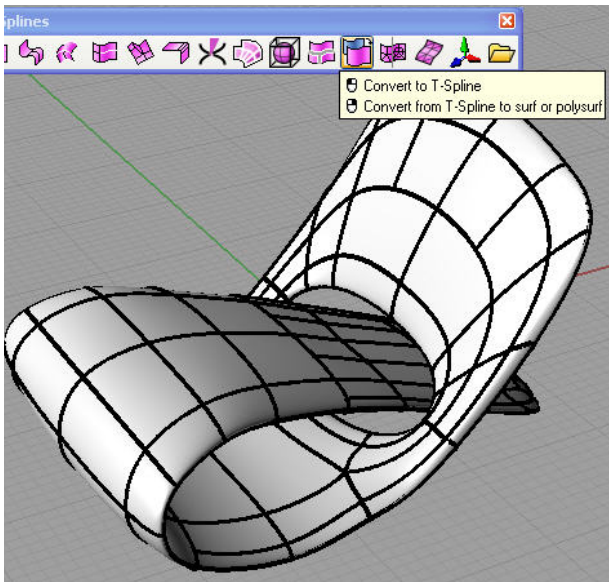
Zebra stripes show an unsmooth surface at tsSetStarSmoothness=1



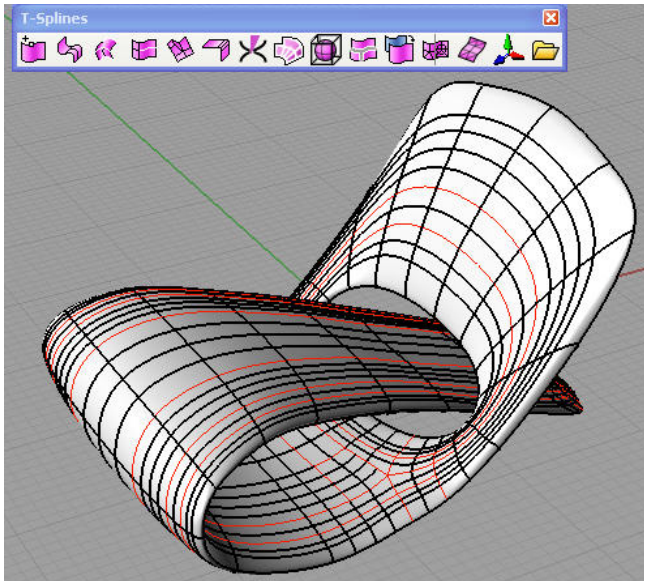
Increasing tsSetStarSmoothness value (to 3, in this picture) yields a smoother surface

7 Convert to NURBS

Converting a T-Splines surface to NURBS polysurf for further detailing or manufacturing is trivial. Simply apply a Boolean or trim operation to the T-Spline, and it will convert automatically to NURBS. The density and number of NURBS patches, as well as the smoothness of the surface around star points, is determined by the tsSetStarSmoothness value.



T-Splines chair



NURBS polysurface chair

Render by Nicolas Denhez

A free trial of T-Splines for Rhino may be downloaded at www.tsplines.com/rhino/