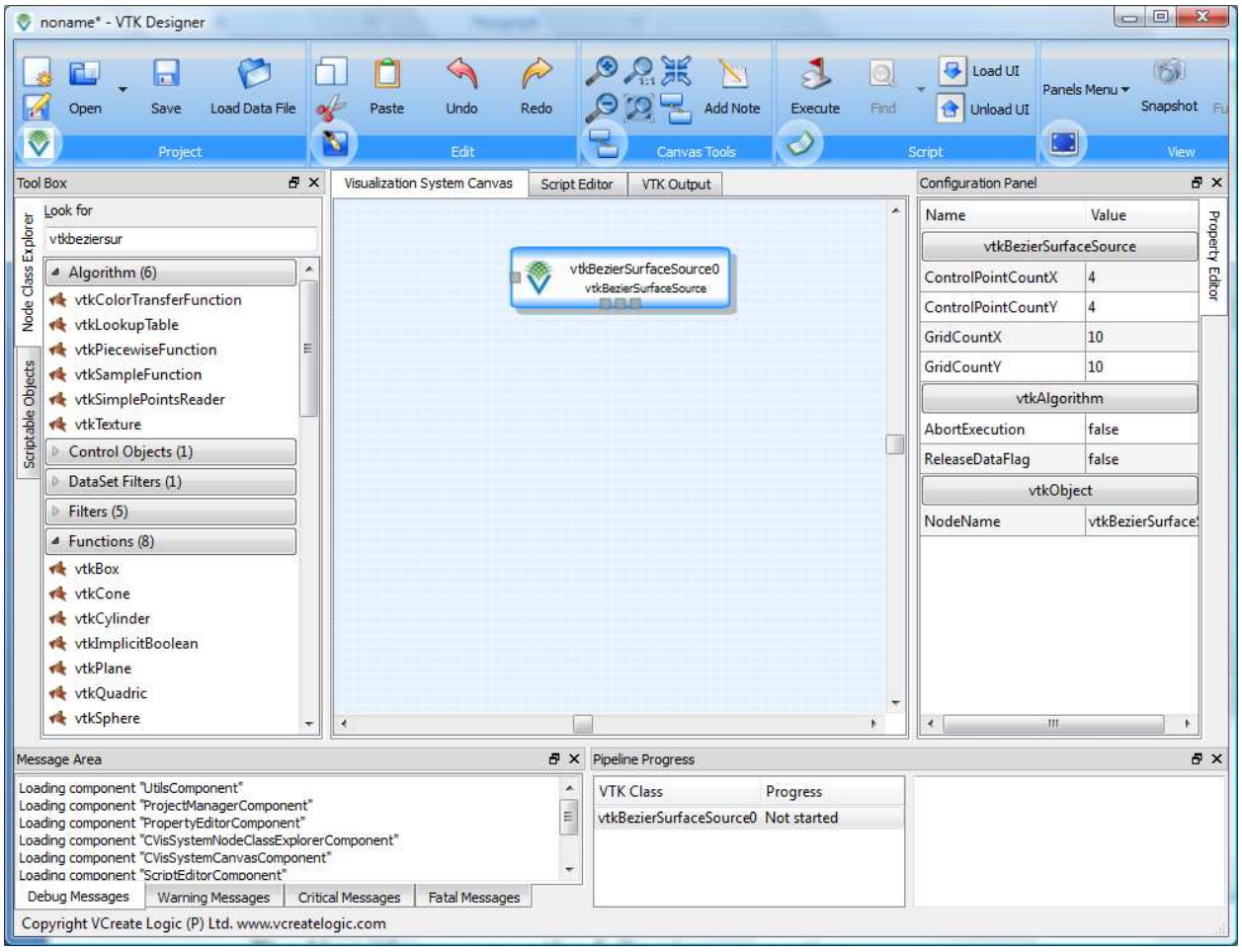
# Bezier Surfaces in VTK Designer 2

*By Prashanth N Udupa (prashanth@vcreatelogic.com), Developer, VTK Designer 2*

***This article doesn't explain the concepts behind Bezier surfaces. You can get all of that from Wikipedia or any math/computer graphics text book. The article explains how you can generate Bezier surfaces in VTK Designer 2 (http://www.vcreatelogic.com/oss/vtkdesigner)***

Start VTK Designer 2 and pull the **vtkBezierSurfaceSource**[1] algorithm from the "Node Class Explorer" on to the "Visualization System Canvas".




## vtkBezierSurfaceSource Properties

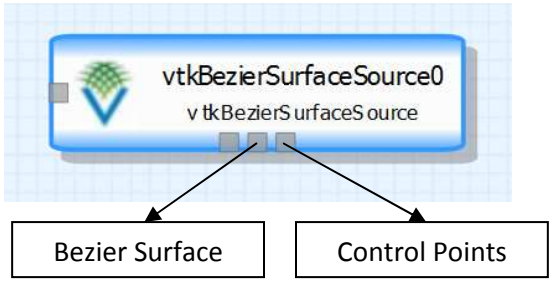
The Algorithm exposes the following properties

[1] vtkBezierSurfaceSource is built using algorithms published by Paul Bourke in his page: http://local.wasp.uwa.edu.au/~pbourke/geometry/bezier/index.html

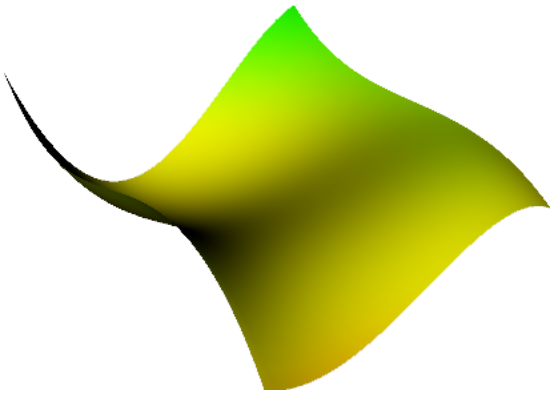
1. ControlPointCountX – Number of control points along X or M axis
2. ControlPointCountY – Number of control points along the Y or N axis. Basically the first two properties provide information about the number of control points
3. GridCountX and GridCountY – These parameters describe the grid on which the Bezier surface will be "super-imposed".

## vtkBezierSurfaceSource outputs

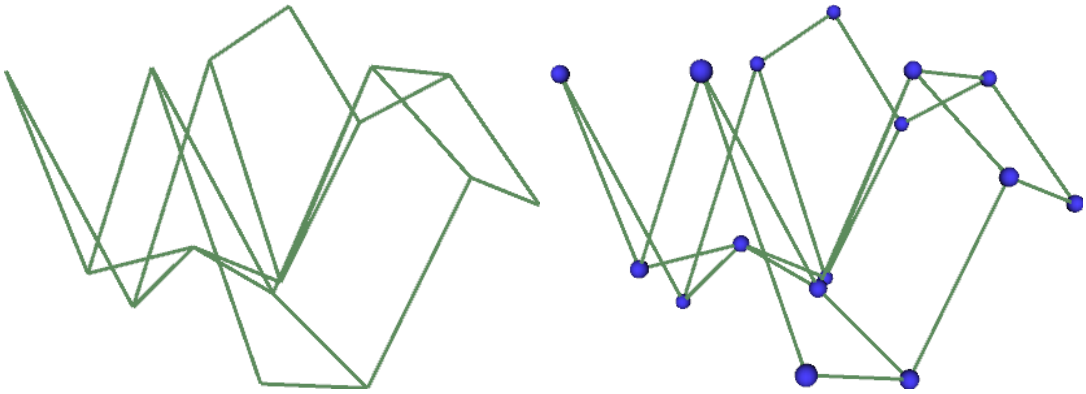
The algorithm provides two types of output




While "**Bezier Surface**" output provides the actual Bezier surface as output, the "**Control Points**" output provides the grid describing the locations of the control points. Both outputs are of type vtkPolyData. For example, the "**Bezier Surface**" output would be something like this

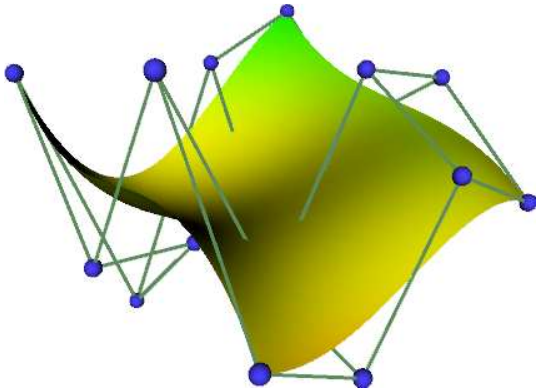


The "**Control Points**" output would look like the one shown in the image on the left.



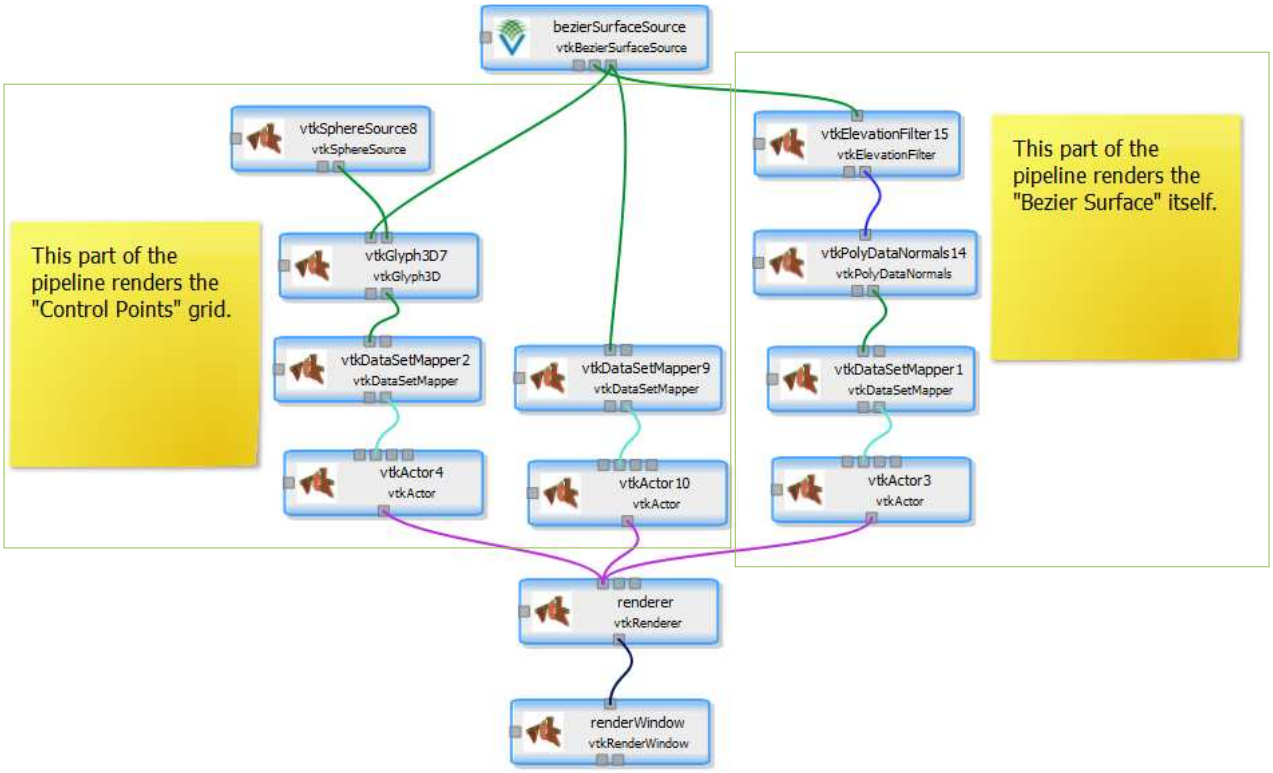
Control points are better visualized by superimposing a sphere glyph at each of its vertex. The image on the right (above) shows the control points with sphere glyphs super-imposed.

When we look at both the outputs together we get



## Constructing a "Bezier Surface" visualization pipeline in VTK Designer 2

To visualize Bezier surfaces in VTK Designer 2, we begin by constructing a pipeline as shown below.




By default all the control points line on the plane z=0. Lets move them randomly in 3D Space. Towarsd this we type the following script in the "Script Editor" tab.

```
function init()
{
    var m = bezierSurfaceSource.ControlPointCountX;
```

```
    var n = bezierSurfaceSource.ControlPointCountY;
    bezierSurfaceSource.resetControlPoints();

    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
        {
            var pt = bezierSurfaceSource.controlPoint(i, j);

            pt.x += VtkMath.randomNumber(-0.5, 0.5);
            pt.y += VtkMath.randomNumber(-0.5, 0.5);
            pt.z += VtkMath.randomNumber(-0.5, 0.5);

            bezierSurfaceSource.setControlPoint(i, j, pt);
        }
    }

    renderer.resetCamera();
    renderWindow.render();
}
init();
```

The above script randomly displaces each control point by 0.5 units on each side, along each axis. Each time you execute the script, you will be able to view different Bezier surfaces. Shown below are some outputs.