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".

- 0 <mark>- X</mark>	U										ner 🖉	VTK Desigr	name* - \	💎 nor
Snapshot View	Panels Menu 🕶	Cript	Q Find	S Execute	Add Note		Redo	Undo Edit	Paste	•	Load Data File	Save	Open	
8		Configuration Panel			VTK Output	pt Editor	vas Scrip	System Canv	Visualization	5 × 1	1		×	Tool Bo:
2 3	Value	Name	*		**************************************		-16						ok for	a Loc
ce	rSurfaceSource	vtkBezier										r	kbeziersur	vt
	ntX 4	ControlPointCoun			BezierSurfaceSource0	vtkBezier						nm (6)	Algorith	ass
	ntY 4	ControlPointCoun			BLC BLC	vtkBez	Ľ				Function	orTransferf	t vtkColo	de Cl
	10	GridCountX										kupTable	t vtkLoo	2 📢
	10	GridCountY						11	vtkPiecewiseFunction			tts		
	Algorithm	vtkA									vtkSampleFunction		objec	
	false	AbortExecution										ure	vtkText	able
	false	ReleaseDataFlag									.)	Objects (1	Control	Script
	tkObiect	vtl							ilters (1)		DataSet	- P		
zierSurface	vtkBez	NodeName										5)	Filters (5	Þ
	TRUCE	Hodenbile										ns (8)	Function	4
											an	ne nder IlicitBoolea ne adric	vtkBox vtkCon vtkCyli vtkImp vtkPlar vtkQua	
	111		, -			(ma)				- 4		ere	vtkSph	-
5					line Progress	× Pipel	8						e Area	Messag
				Progress Not started	K Class BezierSurfaceSource0 I	VTH VTH	ages	Fatal Messa	mponent" Messages	ent" ht" plorerCon onent" Critical N	mponent" ManagerComponer tyEditorComponer stemNodeClassEx stemCanvasComp ditorComponent" ning Messages	ent "UtilsCo ent "Project ent "Propert ent "CVisSys ent "CVisSys ent "ScriptE es Warr	g compone g compone g compone g compone g compone g compone g Message	.oading .oading .oading .oading Loading Loading Debug
	<u></u>		•	Progress lot started	line Progress K Class BezierSurfaceSource0 1	× Pipel	ages	Fatal Messa	* mponent* Messages	ent" it" olorerCon onent" Critical M reatelogi	mponent" ManagerComponer stemNodeClassEx stemCanvasComp ditorComponent" ining Messages (P) Ltd. www.vcr	ere ent "UtilsCorent "Project ent "Project ent "CVisSys ent "CVisSys ent "ScriptEr ess Warr eate Logic	e Area g compone g compone g compone g compone g compone g compone g compone g message g Message	Messag Loading Loading Loading Loading Loading Loading Debug

vtkBezierSurfaceSource Properties

The Algorithm exposes the following properties

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

- 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 "<u>Bezier Surface</u>" output provides the actual Bezier surface as output, the "<u>Control Points</u>" 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++)</pre>
    {
        for(j=0; j<n; j++)</pre>
        {
            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.

