README

This lab assignment included creation of a Utah Tea Pot and providing additional functionalities in order to understand Bezier curves. The flow of the code added is as below:

- MyViewer object – A myviewer object is created which reads the control points from the Utah teapot file and initializes the screen for user interaction.
- Build Teapot – A method is called when the user clicks on the Teapot option. This builds the teapot using the bezier method to create curved surfaces. The resolution can be modified by the user using the button in the menu.
- Build Control Polygons – This method is called when the user clicks on the control polygon button. This method constructs a cage around the teapot depicting the control points.
- Show Normals – This method is called to draw normal over the teapot. These normal are the surface normal at each point.
- Manipulate Control points – Control points can be modified by the user to change the surface of the teapot. This is achieved by making use of the SnManipulator of the Sig APIs.

METHODS ADDED

BUILDTEAPOT(NORMAL_FLAG,RESOLUTION)

This method takes in two parameters normal_flag and the resolution. The normal_flag indicates if the normal need to be drawn along with the teapot. This is decided by the user input. The second parameter indicates the resolution of the teapot. This number is used to divide the single Bezier patch into grids. The smoothness of the surface of the teapot depends on this resolution. The point of a single patch is computed using the below equation:

$$p(u,v) = \sum \sum B_i^n\,(u) B_j^n(v) K_{ij}$$

The points u and v is the resolution passed as a parameter to this function. Once the points are computed using the equation above triangles are pushed in order to render a teapot. If the normal_flag is set it calls a different method makeNormals to add the normals.

MAKENORMALS(GSMODEL,RESOL)

The GsModel containing the vertices of the teapot are received by this method in order to compute the normal. The normal is calculated by considering the vector formed by the current point and the next point in the u direction and a different vector formed by using the current point and the next point in the v direction. These vectors are considered to be approximate tangents to the surface at this point. The cross product of these two vectors gives the magnitude of the normal vector. This is used to push the line for the normal at this point.

buildControlPolygons(started_by)

This method is used for two functionalities. One is to build a cage around the teapot depicting the control polygon or add SnManipulator boxes around the teapot to provide a means to change the control points. The started_by parameter passed to the function. If the value of this parameter is DRAW enum then SnLines are pushed on to the scene to create a control polygon cage. Else snManipulator is added at each point and a callback method is registered for any event changes.

callBack

This method is called whenever snManipulator boxes are moved by the user using mouse clicks. This method identifies the control point which the user is trying to manipulate and calls another method to change the respective control point to the new point.

set_new_control_points(patch, i, j,GsPnt)

This method changes the control points identified by the parameters patch, i and j to the new point given in the parameter. Following this the teapot is again rendered using these new control points set.

Performance Metrics

The following table summarizes the performance metrics for rendering the scene. The table indicates that the code is well optimized compared to the smooth API.

Metrics	Mac				i3			
Resolution	10*10	30*30	50*50	100*100				
Number of triangles used	5696	55616	156736	633536	5696	55616	156736	633536
Time taken to render teapot	0.0032	0.0267	0.0634	0.2749	0.01631	0.1418	0.3949	1.5718
Time taken to render normals	0.0006	0.0063	0.0196	0.0736	0.00216	0.0192	0.05597	0.2301
Time taken to render control polygons	0.00179124				0.0055			
Time taken to render manipulator	0.00184584				1.57741			
Number of lines in control polygon	768				768			
Time taken to smooth API	0.0247	0.4621	2.8096	NR	-	-	-	-