User's Manual for Programming Assignment No.3
For course CS6670, fall 2007

Submitted
By
Leena Kora.
(**Uid: 0527667**)

Abstract

The main aim of the assignment was to produce a graphics program, which helps the user to view 3D polynomial B-spline surfaces using isolines. The user should be allowed to open any 3D polynomial Bspline surface dataset file and display isoparametric curves at nodal values and Knot values in each direction.

The second part of the assignment was to create surface that interpolates to a grid of data with nodal interpolation. The user should be allowed to read grid data from file and allowed to select degrees of interpolation in each direction. Using the selected degrees of the interpolation, appropriate knot vectors should be created and using in the interpolation of surface.

Implementation

About the logic (Algorithms used)

1) I have used the recursive method of evaluating the 3D Bezier curve, first derivative and second derivative, which is given the text book.
It is very simple. It includes copying all the control polygons in two temporary lists and using them in the calculation of curve points using two for loops and storing the evaluated point in a final list. Using the first temporary list we can calculate first derivative values and using the second temporary list we can calculate second derivative values. Evaluation function is called for each value of 't'. I set 't' to have 200 values from 0 to 1 and rendered the curve points using opengl command "GL_LINE_STRIP".

2) To Display frenet frame and osculating circle, I calculated Binormal, Normal, Tangent, and capa(k) at each t value. I have the function called DrawFrenetFrame. The

osculating circle's radius is set to (1/k). It is placed at the given t value using TNB matrix.

3)I have created the evolute by adding point at t value with (1/k)*Normal. I calculated its points in the evaluate function itself.

4) To implement file reading and storing, I have used fstream of c++ language and have written the code to read the files with the specified format.

5) I have set the things to work only with 3D bezier curves and not for 3D rational Bezier curves.

About the GUI (Graphical User Interface)

1 I have used the template, which was provided to us to manage and organize the GUI well. I have created two classes called 'Bezier' and 'RationalBezier', for creating the curves, which are inherited by 'PolyLine' class.

2) As per the condition, I have used C++ for the programming. As I never worked using "FLTK", I spent a lot of time coding for the GUI and also was very curious to code using it.

3) I have added some toggle buttons for building the GUI, so the user has to make sure that, once it's done with the button, it should be toggled off.

4) I added three radio buttons to represent the two types of curves (2D Bezier curve and rational 2D Bezier curve, and 3D bezier curve). As the main purpose of the assignment is to render "Bezier Curves", I have coded such that at least one of them has to be on.

5) I have added one more toggle button to toggle the display of control polygon of the currently displayed curve. The display of the curve can also be toggled on and off.

6)I have also added toggle buttons for adding and deleting points of the control polygon. Users have to toggle them 'on' to add or to delete the points. I have made these buttons to behave as radio buttons because either a point can be deleted or a new point can be added. The user has to toggle them 'on' and has to click on their desired position on the screen to add or remove points.

7)I have made the display window to show nothing so that user can start adding points and can create their own curves, but with toggling at least of the curves display button "on".

8)User can move the control polygon points by just dragging, pushing and leaving. Just make sure that no add or delete buttons should be 'on'.

9)Users can also open any curve dataset file(with the mentioned format) by clicking the "File" menu and selecting "Open" option.

10)Users can also save the curves by selecting the "save" option from the "File" menu. When the 'save' operation is triggered, a dialog box is opened and the user has to enter the name of the file.

11)Users can clear the curve (also the display screen) by selecting “Clear Curve” option from the “file” menu and they can start creating a new curve by adding points or by reading the points from a file.

12)The “Exit” option from the “file” menu can close the application.

13)The users have to toggle off all the ‘add’ and ‘delete’ buttons before using any options from the menus.

14)In case of rational 2D Bezier curve, user can change the rational value of a point by toggling ‘on’ the button labeled as “click to change the rational value”. The user can set the rational value with the help of slider placed at the left of the GUI. I have placed a text display object which displays the value set in the slider. When the user clicks on any point of the control polygon, the curve is updated, leaving the control polygon unchanged. When the rational value change toggle button is set on, no other add or delete buttons should be on. Just to make the GUI friendly, I have set all the ‘add’ and ‘delete’ buttons to toggle ‘off’, when the rational value change button is set ‘on’. But the user has to make sure that that button should be toggle off after done or should not toggle any other buttons, when this button is being used.

Additional GUI menus

15)I have put sliders for changing t, dt, tmin and tmax values.

16)I have used a help text to help the user at times. The 'add', 'delete', and change weight buttons only work when we are using 2D curves.

17)I have put check boxes to display center of curvature curve, line segments between original curve and derived curves, frenet frame and osculating circle.

18)I have used check box to help user do some cool animation to display frenet frame and osculating circle.

19)The user can rotate the 3D bezier curves by dragging the mouse.

20)Scrolling the mouse gives zoom editing.

21)The single stepping is done by pressing 's' key and pressing 'q' stops single stepping.