HomeWork 3

Curve Modeling

Fall 2012

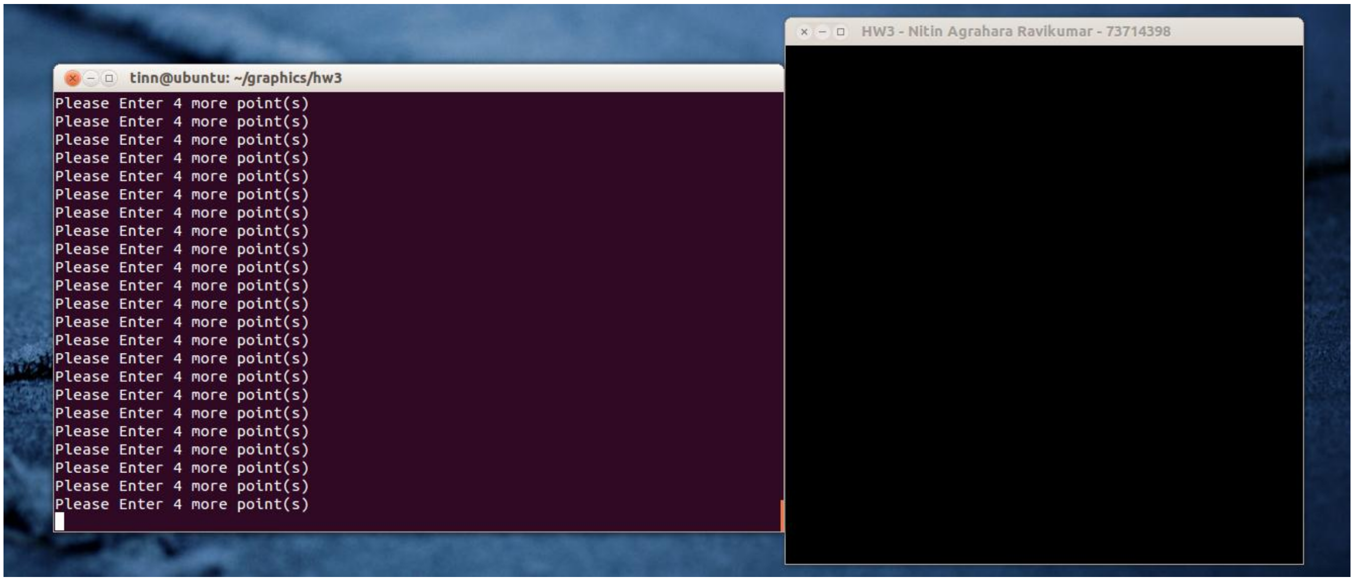
Submitted By -

Nitin Agrahara Ravikumar

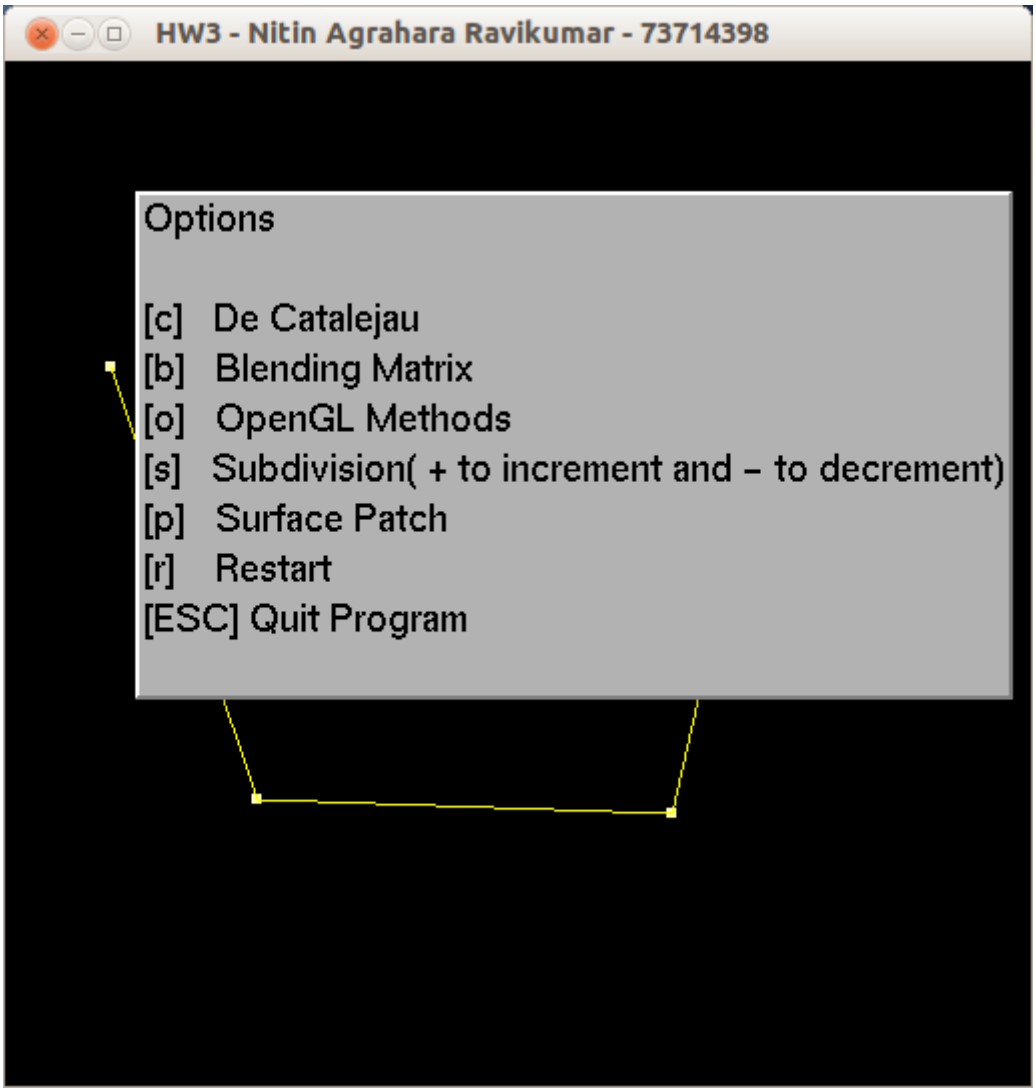
73714398

Structure of the Program

The program has five different parts, one each as specified in the homework. The program starts with a blank screen and the user has to input at least 4 point for a curve to be drawn. This is because the assignment requires us to draw a C1 Bezier curve. The user can input these points by simply clicking anywhere on the output screen. Whenever a point is selected, it will be shown. After four such points have been provided, the control polygon is drawn.



From here on the user can switch between modes. There are two ways of doing these, either by pressing appropriate keys on the keyboard or by using the menu. A menu is generated when the user does a right click on the screen. The options for the keyboard are also shown on the menu. Upon selection the corresponding part of the program starts executing.



The blending matrix and de Casteljau implementations are progressive, i.e. the curve modeling is shown step by step for each increment. Thus the execution snakes through the curve. The OpenGL method on the other hand displays the final result. The subdivision algorithm does the same; it shows the current generated state for the number of subdivisions. The number of subdivisions can be controlled by the '+' and '-'characters on the keyboard.

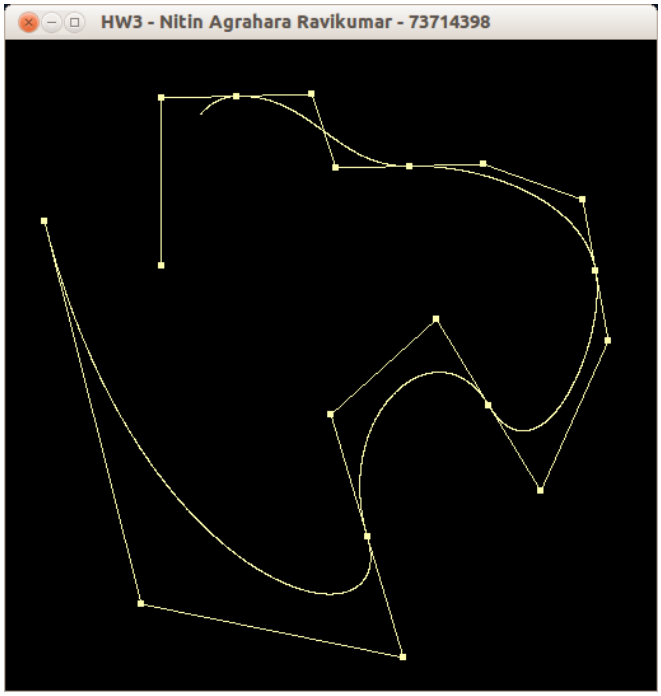
For the implementation of all parts, it is assumed that the curve has to be C1 continuous between different segments. Thus whenever a new point is added to the display, to make the curve resulting continuous, the midpoint between the last two points before the new point was added is considered as a new point. This ensures that the new segment generated has at least 3 points.

When needed the last point is itself considered as a fourth point thus making two complete segments. By considering the midpoint we make sure that three points are collateral thus their vectors are parallel and hence ensuring that the curve generated is continuous.We are therefore able to generate a continuous curve no matter what point the user gives as an input.

Each of the implementations with snapshots is discussed below

Blending Function

Control polygon 1 with 5 parts



Control polygon 2 with 6 parts

Two different curves are shown above. As it is seen, both of them are continuous.
The implementation involves the multiplication of the blending function which includes variables of the order of 3 (since this is a C1 assignment) Thus for every new point, a bunch of calculations have to be made involving several variables of different orders.

De Casteljau

Control polygon 1 with 5 parts

Control polygon 2 with 5 parts

The De Casteljau implementation is a improved over the blending functions. As discussed in the class, the method for calculating a point on the curve is a simple matter of 2 level interpolations. It does not involve variables of different orders. The diagrams show how the interpolation is actually done. This method is computationally more efficient that the blending functions.

OpenGL Method

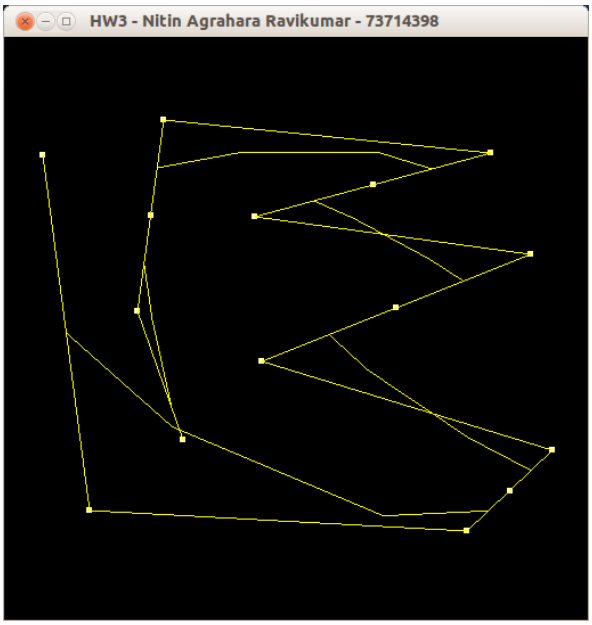
Control polygon 1 with 5 parts

Control polygon 1 with 5 parts

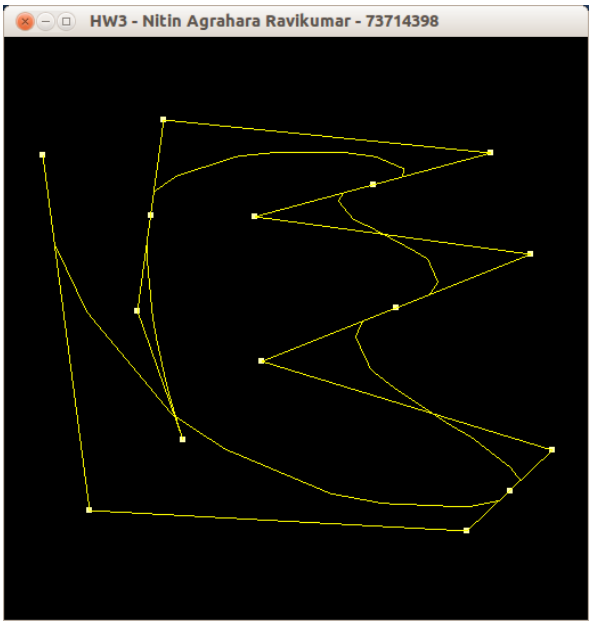
As discovered in the implementation of the Graphics pipeline in the second homework, our standard algorithms no matter however efficiently written cannot be faster than the OpenGL methods because OpenGL has hardware support. We do not have such low level access while writing a program. Hence it is the most efficient method of the lot.

Subdivision

Control Polygon 1



Subdivision Level 1



Subdivision Level 2

Subdivision Level 2

Control Polygon 2

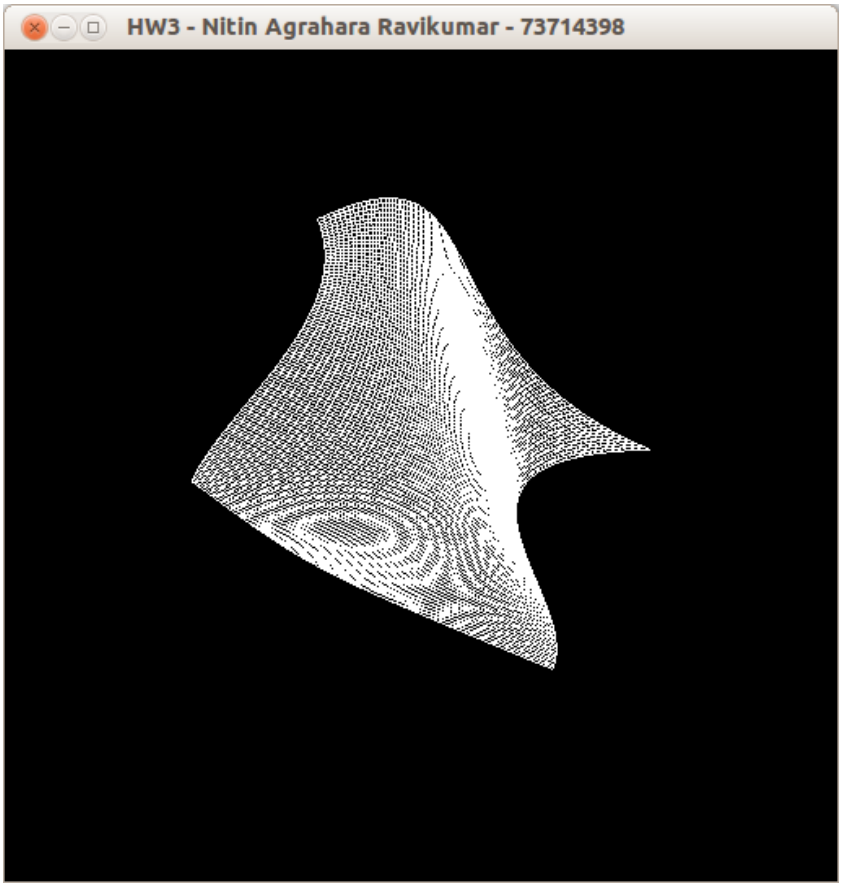
Subdivision Level 1

Subdivision Level 2

Subdivision Level 3

The subdivision is another method of drawing the curve and originates from the De Casteljau algorithm. Instead of interpolating the values on by one, just draw the control points obtained by one level of interpolation and perform the same process for the next level of control points. Each such division gives four left and four right control points. Recursively applying the same techniques and drawing lines between them yields the curve itself. As shown in the above two sets of diagrams, the curve itself is obtained in three steps. This method of curve drawing beats the other methods, but still OpenGL methods remain the fastest.

Bonus - Surface Patch



The above diagram shows a Bezier Surface patch. The number of divisions in the curve has been purposefully kept low so as to enhance the curve view of the surface. A higher sampling will show a completely white object.

Conclusion

Computationally speaking, the OpenGL methods provide the fastest rendering of the curve. But in a mathematical way the best approach to understand how the curve is built, the blending functions are optimal. One level of abstraction above that leaves us with the De Casteljau and subdivision methods. These are advantageous when the curve is to be traced and have great aesthetic value especially in teaching. Disadvantages of these methods as stated previously are they that are computationally expensive.

It was mentioned earlier that the program uses a repeat of points strategy. It is observed that if last two points in a polygon are repeated, the end result does not vary. In case other points in the polygon are repeated such as the first and last ones, the curve terminates at the end. Repeat of two middle vertices yields the same result as the first case.

Acknowledgements

The OpenGL methods used in this assignment have been implemented as described by the '**OpenGL Programming Guide**' found at:

http://www.glprogramming.com/red/index.html