Daniel Son

Progress Report 1

Program Usage Guide

# COMPILING AND RUNNING

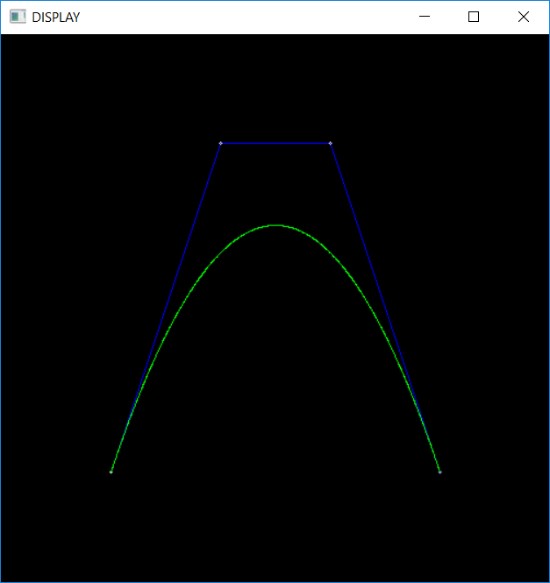
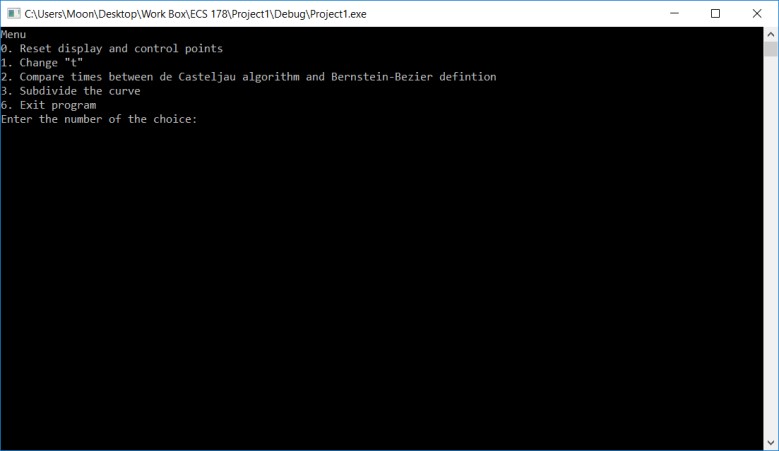
The program is created using FreeGLUT in Visual Studio. Otherwise, it can also be compiled through CSIF using the compile command “g++ main.cpp -lglut -lGL”, and the corresponding display window can be displayed using “Xming”.

# DEFAULT INPUT

The program gets its default input control points through a .txt file called “points\_input.txt”. While it is not necessary, it is an easy method to load up specific points without manually interacting with the display each time.

# USING THE PROGRAM 1. Menu and Display Windows

The program displays an interactive display window and a menu window.



Display Window Menu Window

The display window will show the curve in green and the polygon in blue. It will also display the control points in white.

# 2. Control Point Manipulation

The display window has a size of 500 x 500. This means that control points can be placed within the domain and range of 0 < x < 499 and 0 < y < 499.

To add points, left click anywhere on the display window that is empty. A new control point will be added to the end of the list of control points.

To move points, hold left click on an existing point and release at a new empty spot. The control point will maintain its place in the list and move its position on the display window.

To remove points, right click on an existing point. The control point will be removed from the list.

# 3. Utilizing the Menu

To call up the menu, press “m” on the keyboard while the display window is active. This will display a menu prompt on the menu window. The user can choose an option by entering a number with the corresponding menu choice and then pressing “Enter”.

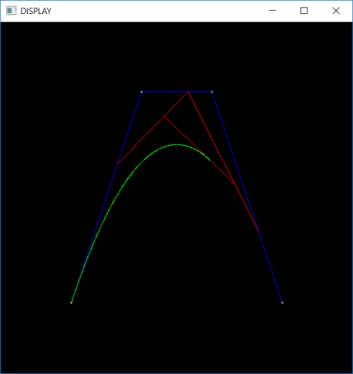
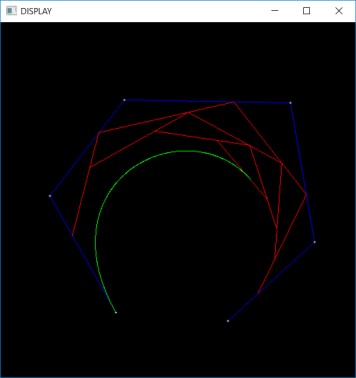
# Menu Functions

# 0. Reset display and control points

Resets the display to an empty state and removes all currently registered control points. Essentially gives user a blank display field.

# 1. Change “t”

Prompts the user to enter a new “t” value for the curve. The new input for “t” must be between 0 and 1. As the “t” value is changed, the intermediate line segments will be shown in red.



**2. Compare times between de Casteljau algorithm and Bernstein-Bezier definition** Runs the de Casteljau algorithm and Bernstein-Bezier definition process to find the points on the curve for current set of control points. Prints out run times for both processes.

# 3. Subdivide the curve

Prompts the user to enter a “t” value to subdivide the current curve at “t”. The polygons will be displayed in blue and purple. The curve will be displayed in green and turquoise.

**4. Exit program**

Exits program.