CSCE 441 - Computer Graphics

Programming Assignment 6

Deadline: Dec. 4th (11:59 pm)

1 Goal

The goal of this assignment is to become familiar with Bezier curves. You will write a program that generates points on a Bezier curve, given the control points.

2 Starter Code

The starter code can be downloaded from here.

3 Task 1

First, download and get the code itself to run. The code has the following characteristics:

- You can specify a file to read as a command-line argument. There is also a default set.
  - 3 sample files have been included. Each one will contain 4 Bezier control points.
- Reading the file is NOT included in the code. You will want to read the file yourself.
- You will want to fill in two routines that are unfinished: `loadFile` and `generatePoints`. The rest of the routine should be done for you.
- Note that points are stored as `glm::vec3`'s.
- The program will display a region from [-10,10] in x, y, and z. You do not need to change anything about the display routine.

4 Task 2

You are to finish the program. This involves filling in two functions (you may write additional functions if you want):

- The `loadFile` routine. It takes in a file name and should return a vector consisting of the four control points of the curve.
  - The first line will be a single number, n, giving the number of control points. Here, the curves you read will always have exactly 4 control points, so n will be 4.
  - The next n lines will each contain 1 control point.
  - Each line will consist of 3 values (x, y, z coordinates of the point)
  - You should read in the 4 points and store them in a vector of points that is returned from the function.
- The `generatePoints` routine. It takes in a vector that contains 4 control points. It should output a vector of points that are along the curve.
  - The input will be a vector of the four control points, in order.
  - You should generate a list of at least 51 points along the curve, in order, from the start of the curve ($t = 0$) to the end of the curve ($t = 1$).
    - You can do this by directly evaluating the polynomial function. However, to get the full credit you need to implement de Casteljau's algorithm.
  - The results should be stored in a vector of 3D points that are returned.

5 Deliverables

Please follow the instruction below or you may lose some points:

- You should include a README file that includes the parts that you were not able to implement, any extra part that you have implemented, or anything else that is notable.
- Your submission should contain folders "src" as well as "CMakeLists.txt" file. You should not include the "build" or "DataFile" folders.
- Zip up the whole package and call it "Firstname_Lastname.zip". Note that the zip file should extract into a folder named "Firstname_Lastname". So you should first put your package in a folder called "Firstname_Lastname" and then zip it up.

6 Ruberic

Total credit: [50 points]

[10 points] - File is read in correctly
[30 points] - Points are generated along curve correctly
[10 points] - Points are generated using de Casteljau algorithm

7 Acknowlegement

The assignment is from John Keyser with slight modification.