Computer Graphics Workshop 2 - Mountains

Goals

1. Understand using OpenGL with SDL
2. Learn about VBOs
3. Understand the basics of shaders
4. Learn the fundamentals of height maps

Problem introduction

For this workshop you will read a height map from a file and display it as a surface in 3D space. Before this workshop you should have the starting point compiled and running on Linux and have read through the source code. The following problems are ordered by difficulty, and should be done in this order. As a general rule, feel free to expand on the shaders and rendering methods to make the terrain look more interesting.

Problem 1 (low difficulty)

Insert the correct calculations in the `loadTerrain` function, transforming the pixels from the heightmap to vertices. Use the `getTerrainVertex` function to fill in the positions array with the correct values. The correct calculations for the normal vertices have been done already as an example. Replace the current assignments to the positions array (ie. `Vector3(x - 256.0f, 256.0f - y, height)`) with calls similar to those for the normals array. If done correctly, you should now have a green mountain range on your screen and you can proceed with the next problem.

Problem 2 (medium difficulty)

1. Make the 'w' key toggle between wireframe and surface rendering modes. There are several ways to achieve wireframe rendering, feel free to look them up. Note that in the current viewpoint, some aliasing will occur and the wireframe render will not look optimal.
2. Make the '+' key create a rougher surface and the '-' key a smoother/flatter surface by applying a multiplier to the height map values. Don't worry about updating the normals, just the positions.

3. Adjust the base color of the surface in the `terrain.fs` fragment shader. You can do a simple color change, or you can use your imagination to create a colorful landscape.

4. Hard code a "snowy mountaintop" into the shader by setting the color of all fragments above a certain height to a different color.

Problem 3 (high difficulty)[Bonus Problem]

In the `update` function, add code that adjusts the `viewpoint` and `viewtarget` vectors responding to some input keys. Pick one of the proposed problems:

- Implement a fixed viewtarget camera which rotates around the terrain. (Low difficulty)
- Implement a freely rotatable and moveable first-person camera. (High difficulty)

Submission

The deadline is at the end of class, but if necessary, they may be submitted by 6pm of the day of the Workshop. Place in a ZIP file the following and submit on the LML Course Manager:

The top level of the zip file should contain a directory called
firstname.lastname.project as described below:

In a directory called "firstname.lastname.project" (e.g. mary.smith.project)
(1) a file named "AnswerJournal.txt" which should list
- Your full name and student ID
- Mention which of the problems you solved.

(2) The source code, Makefile (the project must compile using "make") and
(3) Working executable of your solution