CS 455
Program #2

Programming Shaders

Dr. Egbert
(100 Points)

The purpose of this project is to help you learn how to program Shaders in OpenGL.

For this program you will draw three versions of a simple house.

First, create the vertices for your house. In addition to the position data, you will need to store color data for these vertices. You can do this in one of two ways. You can either store all of the data in a single array (x, y, z, r, g, b for each vertex), or you can create two separate arrays - one array for position data and one array for color data. You will then need to create the VBO(s) and VAO to be used in your shader code, then connect the data to the shaders.

You will then need to render your house three times, altering the location, size, and color of the house each render. You will use the same vertex data for each of the three drawings, but you will use transformation matrices to alter the locations and sizes of the three houses.

To alter the location and size, you will need to set up a transformation matrix in the application code that you pass to the vertex shader via a uniform. To alter the color, you will need to set up a color mask vector in the application code that you also pass to the vertex shader via a uniform.

Setting up the transformation matrix and color mask vector is easily done using glm, which is a math library designed to be used with OpenGL. It includes routines such as translate, rotate, and scale for setting up transformation matrices that can be used in your code.

Another routine you will need to use is glm::value_ptr. This routine creates a pointer to your data (matrix or vector), to be passed to the uniform.

Also remember that OpenGL uses matrices in column-major order, so you will post-multiply your transformation matrices by your vertices, which is probably different than the way you learned in CS355.

Once you have set the two uniforms, you will then render the house.

In the Vertex Shader, you will need to input the vertex position and color data. You will also need to declare a uniform for your transformation matrix. You will then use the uniform you pass in to alter the position data prior to sending the data to the fragment shader.

In the Fragment Shader, you will need to input the color, and you will need to declare a uniform that you use to alter the color of the object being passed into the shader. You will then output the altered color to the framebuffer.

Once you have completed this assignment, submit your main routine, your shader code, and a screen shot of your output through LearningSuite.

Your output should look something like this (although yours may differ based on your transformations and color changes):