COSC 4370 – Homework 3

Name: Zitlali Silva

March 20, 2022

1 Problem

The assignment is to implement the 3D viewing and Phong shading model. I need to complete the GetViewMatrix() function in Camera.H and the projection matrix in main.cpp. Along, write a code the vertex and fragment shaders for the phong model to shade a cube.

2 Method

There were a few functions that needed to be modified in the provided code: main.cpp, phong.frag, phong.vs and Camera.h. It was not needed to edit the whole part of the code, only a very small portion of it. The geometry part in making the cube in constructed in main.cpp. The stubs for the shaders in the phong.vs and phong.frag. In order to view and move the cube, it will be in the GetViewMatrix() function in Camera.H and the projection matrix.

For the assignment, it is to learn how to make a 3D model and practice the different types of shaders such as ambient, diffuse and specular.

3 Implementation

3.1 main.cpp

In the main.cpp, I had to set up the project matrix. Here, we will implement the vertical field of view in radians, zooming and the camera lens. A vertex that happens to have x= 0 and y= 0 should be at the center of the screen. However, we must use the z coordinate as well to get the 3D affect. Here it has a 45-degree field of view, ratio, display range of 0.1 to 100.

Reference: Tutorial 3 : Matrices (opengl-tutorial.org)

3.2 Camera.h

I had to do the GetViewMatrix(), this will return the view matrix calculated using the Eular angles and the lookAt matrix. This function will let us move around the scene. In order to define a camera, we need the position, direction, and a vector pointing right and upwards. Along, we will be using the LookAt function that requires a position, target and a up vector. We make three (float) variables: radius, Camera X (CamX = x coordinates) and Camera Z (CamZ = z coordinates) and then providing the LookAt function.

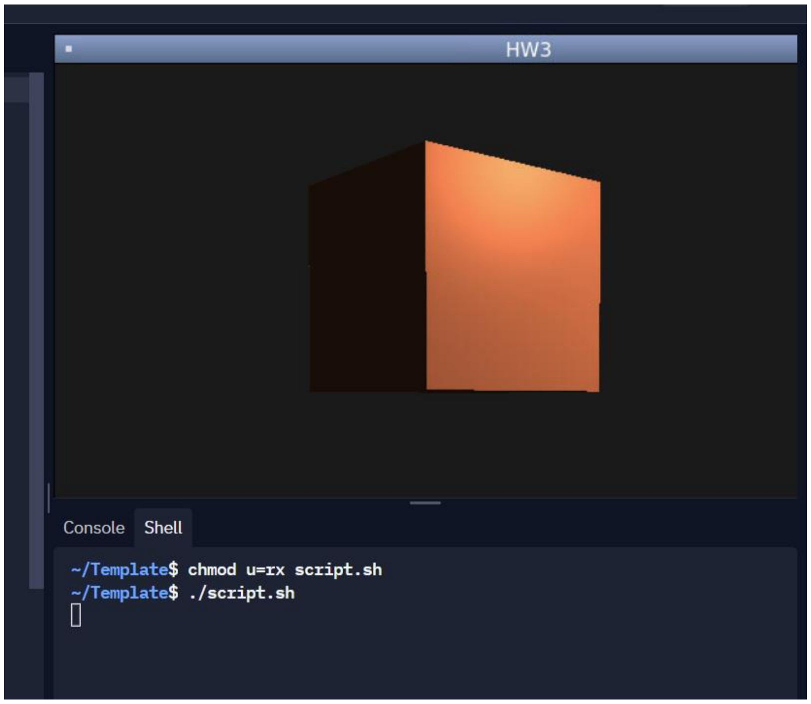
3.3 Phong.frag and Phong.vs

In phong.frag and Phong.vs we are doing the shaders and the lighting for the cube. The three major components of phong lighting model are ambient, diffuse and specular. For ambient, we take the light color and multiply it with a small constant factor, next multiply with the object color and use it as fragment color. Diffuse simulates the directional impact a light object has on an object. Here we declare it as a uniform, then update the uniform in the loop and use lightPos vector. Then, we multiply the vertex position with model matrix. Lastly, we add them all together.

In phong. Vs, Shaders begin with a version declaration following with a list of input and output variables, uniforms and main function.

4 Result

After implanting and modifying the code in the files, we can say we made a 3D figure and having the keys functioning as well.



```
~/Template$ chmod u=rx script.sh
~/Template$ ./script.sh
```