COSC 4370 – Homework 3

Dean Ferreira (PSID: 1463211)

October 24th, 2021

1 Problem

The assignment involved 3D viewing and a deeper dive into OpenGL by implementing the Phong shader model. There are features that have been included such as the ability to pan and shift the camera, the mouse will rotate the camera and using WASD will shift the camera.

2 Method

To view the object from the camera, GetViewMatrix() and the projection matrix must be completed. The vertex and fragment shaders must be written for the Phong model to shade the cube, doing everything correctly will reproduce an image like the one provided. Using the camera documentation provided on Learn OpenGL, we use the Zoom method to establish the field of viewing allowing for the scene to be set. To return the view matrix, we use the LookAt function, this will allow us to control the camera's movement.

3 Implementation

Beginning with the implementation in main.cpp, we are setting up the perspective for our scene. The field of view defines how much of the scene is visible, this is how we will implement the zoom ability. When the field of view becomes smaller, the smaller space is projected over the same coordinate space giving the illusion of zooming in (LearnOpenGL). In Camera.h we complete the function GetViewMatrix() that returns the view matrix calculated using Euler Angles and the LookAt Matrix. Euler angles are 3 values that can represent a rotation in 3D, the 3 Euler angles are pitch, yaw, and roll (LearnOpenGL). For phong.frag and phong.vs, I used Basic Lighting from LearnOpenGL as a reference. The are three main components when it comes to lighting, ambient, diffuse, and specular. When these components are combined this gives us the Phong model. Using a combination of each component we can create a color that aims to mimic how lighting in the real-world works. The main goal is the show where the lighting is aimed, and that area should be the bright and it will slowly fade into the original color of the object. The Phong model represents how power shaders are and how they can calculate how lighting affects the fragment's colors (LearnOpenGL).

4 Results

The output of this program was a live scene of the cube, I say live because the program accepted input from the user that allows them to control the camera. Using the mouse and the WASD keys, the user can view different angles of this scene.