Osinor Igwemoh

Report for Homework 3.

Problem Description

This assignment builds on our previous knowledge of openGL and introduces us to the concept of shaders. We are required to include some more openGL libraries to create color and draw images using shaders that we write. The model provided in the starter code of the homework files is a red colored cube and we as students are tasked with completing the camera class using Euler's angles to calculate view matrix and render the model based on mouse movement and keyboard strokes. The assignment also requires us to complete the projection matrix for the model and build the phong models which are our vertex shaders and fragment shaders.

Algorithm / Method/ Implementation

For the getViewMatrix() function, the assignment requires that we utilize a function that can change based on the input from the keyboard and the mouse so as to render the model effectively. I used the LookAt() function that takes the position, target and up vectors respectively which will be multiplied by each vertex in the vertex matrix to produce the required view. Using the three Euler's angles; pitch, yaw and roll the rest of the starter code was able to take care of the camera effects.

The second major task was completing the projection matrix. The projection matrix describes the angle we see the model from amongst other things, it can be orthographic or perspective but I utilized the perspective projection Matrix for my assignment and I used the perspective function provided by the glm library which takes the angle as an argument (45degrees) and then the aspect ratio as a second argument which I used 4.0f/3.0f for and the Near clipping plane as the 3rd argument and far clipping plane as the final argument. This function creates a 4 by 4 matrix which can be multiplied by the vertices to create the required projection at every render point.

The final and most interesting part of this project assignment was completing the phong and shader models, the starter code had already done the bulk of the work for this part of the homework, all the necessary values were already queried from the main file and we just had to use them in our shader programs that ran on the GPU, I was able to learn so much about how graphics work on the GPU because of the number of calculations required when trying to rasterize an image.

The shader models have two different files, one was the vertex shader which described the points on the image and the other was the fragment shader which for the most part defined the color / lighting of the image. The phong model is a combination of three different types of lighting, the ambient lighting, diffuse lighting and the specular lighting. The ambient lighting gives the object a general light as if there was just light from all directions in the room where the object sat, the diffuse lighting simulates directional light as if the light of a flashlight was shown on one side of an object and the specular lighting tries to create the effect of an actual light source that is precise shown across the surface of the box model in this case. OpenGl makes it very easy to accomplish these effects using the shaders.

The ambient lighting was calculated by the product of the ambient Strength and the color of the Light provided by the starter code and this result was multiplied by the object Color to render to the model. The diffuse lighting comes from an angle and this is where the normal vector comes into play. The light direction is calculated by the difference in the light Position and the Frag Position and the maximum between zero and the dot product of the normal and the light direction is used to calculate the diff variable which is then multiplied by the light Color to get the diffuse variable which is added to the ambient from before and multiplied by the object Color to get the final result rendered to the window.

Specular Lighting is what brings it all together, it calculates the view direction as the difference between the view Position and the frag position. The spec is calculated by raising the power of the maximum between the dot product of the view direction and the reflect direction and zero to 32.

Then the variable specular is calculated by taking the product of the specular Strength of 0.5 and spec variable and the light Color. Then the specular variable is added to the result variable which gives the final result as desired by the assignment specifications.