Garrett Dunn

17th April 2022

CS 330

SNHU (Southern New Hampshire University)

Final Project Reflection

The idea for this project was to keep everything simple enough to help learn the fundamentals of programming using the OpenGL libraries. The scene I decided to create was my computer desk. It contains my desktop, my computer monitor (which acts as the complex object), and my two desktop speakers. I decided not to build out the whole desk and just kept the top as a plane because during the first few modules I had a tough time comprehending OpenGL. By keeping the scene simple, I was able to learn how to creatively program my scene and this simplicity played a huge part in my success in understanding the content of this class. Although the scene looks simple initially, the code was designed with modularity and functionality in mind.

When implementing the virtual camera for the scene, I relied heavily on the module examples in the CS-330 master file that was supplied to us. I knew that I wanted the user to have complete control over what they could do in the scene, and utilizing those examples gave me the opportunity to implement the camera mechanics that I have in my project today. The camera itself is a global variable and declares the initial world position of the camera, timing, and starting projection type. The camera variable is called in the main loop to process the inputs from the keyboard, which allows for movement, changing the projection type, and the light testing that helped me position the lighting in the scene. By utilizing callbacks, we can look around the scene based on mouse movements, change the speed at which you navigate the scene using the scroll wheel, and display which mouse buttons have been pressed and released in the terminal. When executing the code, you can see that the control of the camera accomplishes everything described above.

This code, as I have mentioned above, was built with modularity and functionality in mind. At times, it was difficult to manage this project as there are so many moving parts, and I would be lying if I said that I was not confused at countless moments throughout the development of this project. By utilizing best practices, and taking a step back at times, I was able to do the best I could to make sure that the code was as clean and modular as possible. There are many custom functions in the code, but I want to focus on the shader refactorization I implemented during module 7. According to many tutorials and forum posts on the internet, it seems like a vertex and fragment shader is needed to be programmed for every single object in the scene. This makes adding new lights or objects to your scene an incredibly arduous task. By implementing a single shader program for both the vertex and fragments of the scene, we can freely add new objects and lights at any moment without having to write a new shader. This solution seems like the best approach from a programming standpoint. The freedom brought by this solution would truly save time if this were to be a real-world, collaborative project.

This class has been one of the most difficult yet rewarding classes I have taken so far at SNHU. I have learned so much about OpenGL, but most importantly I have solidified my problem-solving and programming skills in a way that makes me feel much more confident about my future. I look forward to continue learning about OpenGL in the future, and I am thankful I was able to get a grasp on this class’s material.