Charles Campbell

Design Decisions & Reflection for OpenGL Project

April 18, 2025

Design Decisions & Reflection

An office setting, where there’s a computer/laptop with a book/notepad and writing utensil (like a pencil) is a common area I work around in. These objects are familiar to me, and I can easily picture their general shapes and textures if asked about them. This is mainly why I chose to recreate this in a computer graphic 3D environment. While learning about OpenGL and developing my first 3D rendered scene, I wanted to pick an setting that I could reliably picture and have experience with; that way when manipulating the OpenGL library with its shapes, lighting, and shaders/textures to practice learning the basics.

In this program I developed, a user can look around with the mouse in the scene, and change its position using the W, A, S and D keys, but they can also utilize the Q & E keys to easily raise and descend the camera for either a more level or birds-eye view on the scene. Additionally, upon pushing the O or P keys, the camera will also swap between an orthographic and perspective displays. Orthographic displays make the shapes and environment appear more 2D, and focus more on the objects scale and accuracy of true size. This is to help give a better sense of scale and depth to the scene, as the user can swap between the two scenes easily with a push of a button. The camera is also capable of it’s movement to be slowed down or sped up with the use of the scroll wheel on the mouse. These functions of the keyboard and mouse input device allow the user a wide range of movement and ability to see each object, and the overall scene, at many different angles and perspectives.

A major custom function I implemented in this program was the ability to make individual functions to create each object in the scene. Instead of cramming multiple instances of objects being created (such as DrawTorusMesh() and all it’s pre-requisite lines of code) within the SceneManager’s RenderScene() function… each object had it’s own unique function such as “RenderComputer()” or “RenderGlasses()”. This helped keep an organized and cleanly commented SceneManager file, and allowed for quick and easy re-utilization of the “DrawShape()” for each object. Instead of having to scroll continuously to search for and find a specific shape if I needed to edit it’s color or texture, finding it and editing would be easy and take much less time. Additionally, if I wanted to remove an object I could simply remove the called function in the RenderScene(), rather than having to dig for the beginning and end of each specific object and then deleting it. It helped immensely in keeping an organized and easy to read code; and a practice that I can and will utilize in the future for not only OpenGL projects, but other programs that would benefit from encapsulated functions.