William Berry

3/2/2025

CS-330  
  
**Reflection on 3D Scene Development**

This document reflects on the design and development of a 3D scene created using OpenGL, focusing on the choices made, user interaction, and code structure. The scene represents a stylized desktop environment, complete with a desk, monitor, keyboard, mouse, teacup and saucer, stacked books, and a multi-tiered organizer.

The selection of objects within the scene aimed to create a familiar and relatable setting – a modern workspace. This familiarity allows viewers to easily grasp the scene's context and focus on the rendering techniques employed. The objects themselves were chosen to provide a variety of geometric shapes, including planes (for the desk surface), boxes (for the monitor, keyboard, books, and organizer components), spheres (for parts of the mouse, teacup, saucer, and vase), cylinders (for parts of the teacup, saucer, and monitor stand), a tapered cylinder (for the monitor stand arm), and a torus (for the vase's rim). This diversity enabled the demonstration of various transformations, such as scaling, rotation, and translation, within the OpenGL environment. Furthermore, the objects were chosen to represent a range of materials – wood, plastic, metal, glass, and ceramic – providing opportunities to experiment with different shader settings for ambient, diffuse, and specular lighting.

User navigation within the 3D scene is accomplished through keyboard and mouse controls. The camera's position can be adjusted, allowing the user to explore the scene from different viewpoints. The Q and E keys control vertical movement, raising and lowering the camera, respectively. The W and S keys handle forward and backward movement, effectively zooming in and out of the scene. Horizontal panning is achieved with the A and D keys, moving the camera left and right. Additionally, the mouse scroll wheel provides control over the camera's movement speed, allowing for both rapid traversal and precise adjustments. This combination of controls offers a simple yet flexible way to navigate the 3D environment.

The underlying code is organized to promote modularity and maintainability. A key element of this organization is the use of classes. The SceneManager class serves as the central hub, managing all aspects of the scene, including object placement, lighting, and rendering. The ShaderManager class encapsulates all shader-related operations, such as loading, compiling, and linking the vertex and fragment shaders. It also provides utility functions for setting uniform variables within the shaders.

Within the SceneManager class, extensive use of helper functions further enhances code organization. Separate functions are dedicated to drawing each distinct object or group of related objects, such as DrawDesk, DrawMonitor, DrawKeyboard, DrawTeacup, and so on. These helper functions encapsulate all the steps required to render a specific object: setting the appropriate shader uniforms (material properties, colors, or textures) and calling the necessary drawing commands. This approach makes the RenderScene function itself very concise and readable, consisting primarily of calls to these clearly defined helper functions. This modularity also makes the code easier to modify and extend. For example, changing the dimensions of the keyboard only requires modifying the DrawKeyboard function, without affecting other parts of the scene. Similarly, adding a new object involves creating a new helper function, without disrupting existing code. The helper functions are reusable; drawing multiple books, for example, simply involves calling DrawBook multiple times with different parameters. The clear separation of concerns between scene management, shader management, mesh handling, and individual object rendering makes the code more robust and easier to debug.