7-1 Final Project: Reflection

Bradley Wells

Southern New Hampshire University

CS-330

Professor Graham

October 14, 2024

Looking back at the proposal for this scene, I can identify some differences in planning versus outcomes from development choices. Some of these choices include picking shapes for the mug, the candle, and adding shapes for details in the scene that I did not previously identify. For the mug I chose to use a tapered cylinder and a torus for the mug’s handle due to the unique shapes. One unexpected choice I had to make was the addition of another torus onto the mug to add depth to the scene and bring out a lip in the object. This was one aspect where I had to add shapes to account for more detail. The candle was my most complex object and I wasn’t entirely sure how I would construct it. After finding my way with manipulating drawn shapes I was able to work through a solution using a variety of cylinders, tapered cylinders, and a torus to reflect the photographed scene well. The variety of tapered and regular cylinders enabled the creation of the uniquely shaped candle. Additional challenges with the candle that presented were creating an additional cylinder to represent the wax inside the glass jar and another cylinder to be the wick in the candle. Both additions added detail to the scene to better recreate it. One development choice that worked extremely well was providing my own textures to best represent the objects in the scene. While it involved resizing and formatting the images to make them best compatible with OpenGL, I feel as though it added a lot in terms of realism to the final product.

Navigating the scene should feel intuitive to the user and offer them the ability to examine all scene elements from many perspectives. The ViewManager class is responsible for this capability. In this scene the user can move left/right (A/D), in/out (W/S), and up/down(Q/E) to move the camera where they want to go. Additionally, the mouse can be used to intuitively change the view of the camera as it flies in the scene using the ProcessMouseMovement function. Preset orthographic and perspective views have been established using the O and P keys offering a 2D flat angle and a 3D perspective view to the scene that the user can snap too. These keyboard inputs are processed in the ProcessKeyboardEvents function. The camera speed can be sped up or slowed down by using a mouse scroll (down/up) to control speed using the Mouse\_Scroll\_Callback. The combination of the mouse and keyboard controls offer intuitive and full control of this 3D environment to the user.

When it comes to code modularity and organization in this scene, the biggest change I made was each object being its own method. This was incredibly helpful to me in the design process so I could turn off methods and focus on one object in the scene to correctly build it without other objects getting in the way. This also offers more scalability so the scene can have other objects easily added or taken away without sorting through what would have been a single RenderScene() method with 500+ lines of code to sort through. Other functions in this code make the scene setup more modular and organized by breaking down tasks into distinct, manageable pieces. The LoadSceneTextures() function handles loading all the necessary textures into memory by using the helper function CreateGLTexture() for each texture. This approach simplifies texture management and allows for easy modifications. The DefineObjectMaterials() function centralizes the definitions of materials, such as glass and ceramic, ensuring that material properties are organized in one place. This allows for efficient updates or additions without disrupting other aspects of the scene. Similarly, the SetupSceneLights() function configures the lighting sources by setting parameters like position, color, and intensity, keeping lighting control separate from other scene elements. Finally, the PrepareScene() function ties everything together by calling these modular functions in sequence, ensuring that the textures, materials, and lights are loaded and set up in an organized manner. This modular design enhances the clarity and maintainability of the code, making it easier to extend or modify individual components without affecting the whole scene.