Alexander Flood

SNHU CS-330

20th April 2025

**Justification – Development Choices**

For this 3D scene, I decided to create a small collection of household objects, including a mug, book, chest, and wine stopper. I arranged the items closely together to create a cozy environment on the desk. The goal was to create a small and believable scene using different 3D shapes and techniques, such as making complex objects out of simpler ones.

The **metal mug** is made of two cylinders for the inner and outer body, joined by a torus to create a smooth drinking rim. Three flattened boxes form the handle, which were then placed on the mug's side. I initially considered using a torus, but this did not respect the geometry of the cylinders and ruined the geometry by protruding through key shapes. The boxes provided a simpler composition and alignment.

The **fabric book** is built from multiple flat boxes of slightly different dimensions to represent the front cover, back cover, binding, and pages. Here, special care was given to make sure the materials for the book cover and pages were not highly reflective or shiny but instead provided a matte finish. The cloth texture paired with low-specular lighting captures the soft and woven appearance of nice book covers. Another method one could take to recreate a book would be using a single box mesh shaped like a book and texturing the sides to look like pages and the front and back to look like covers, but the model would be overly simplistic, losing depth and realism.

The **metal cork stopper** uses a cylinder for the main body with a metal texture and material. The metal material provides strong specular properties that drive the reflective properties of polished metal, creating sharper highlights and subtle reflections, just like real metal surfaces. Then, I placed two black tori of varied sizes at two different points of the stopper. A dark rubber texture and material is applied to simulate the non-reflective properties of dark rubber, which should not respond strongly to direct lighting and instead appear matte and diffuse.

The **candle** is modeled from a single cylinder with colored shader for the wax body. A larger cylinder with a glass material and color enclosed the wax, and then a smaller cylinder wrapped in two tori were also given a glass material and color to simulate the neck of the candle. Materials were carefully chosen with a low specular and non-reflective material used for the wax and wick to simulate their real-world counterparts. Meanwhile a more reflective material was used for the glass to interact with the scene lighting and catch highlights, standing out as my favorite object in the scene.

The **wooden chest** uses large rectangular boxes for the top and bottom to form a container. This form was enhanced by applying a complex texture to the top of the chest, where a texture emphasizing the seam is applied to the sides of the box, while a texture without the seem is applied to the top. The leather straps are represented by narrow and long boxes with a leather texture and material applied to reflect the subtly reflective properties of leather. Finally, small boxes of various shapes were used to represent the metal latches on the front of the chest. A reflective property was applied to the latch to provide proper reflections and highlights to simulate metal. Another way to simulate or emphasize a seam between the two parts of the chest, a very slight rotation could be applied to the top of the chest, but further 3d modeling would be needed to making hinges to maintain realism; without hinges, the seam would appear, but the chest would lose realism.

To make the scene flexible, I anchored each object to a central base position and built all its components relative to this point. This made it simple to reposition entire objects during the scene layout without breaking their alignment. For the book and chest specifically, I also rotated local offsets, allowing me to rotate these objects without breaking alignment between objects.

**Navigation**

Navigation in my 3D scene is designed to give the user liberty to explore the scene through both keyboard and mouse inputs. This reflects a common pattern used to navigate many first-person digital environments.

Keyboard controls are handled in the ‘ProcessKeyboardEvents()’ function. This function allows users to move the camera forward and backwards with keys W and S, strafe left and right with A and D, and move vertically with Q and E. Each key press calls g\_pCamera->ProcessKeyboard() with the respective direction.

Mouse input is captured by the Mouse\_Position\_Callback() function. This tracks the mouse of the user and maps the offset of the mouse position between frames to the yaw and pitch of the camera. Allowing the user to navigate in this manner simulates how players typically look around in 3D or game environments.

The scroll wheel actions are handled by the MouseScrollCallback(), which provides a second method of control by adjusting the movement speed of the camera. Scrolling up increases the speed in which the camera moves, while scrolling down decreases it. This callback allows users to dial in their movement speed to their tastes, allowing everyone to navigate the environment comfortably.

To support different camera perspectives, I implemented the ability to toggle between perspective and orthographic projection modes. Using the keyboard, the P resets the camera to a familiar perspective view, while pressing O activates orthographic projection and cycles through three preset orthographic angles (ORTHO\_FRONT, ORTHO\_SIDE, ORTHO\_TOP). The transitions are managed by tracking a private member variable which holds the current orthographic position and updates it accordingly.

To prevent accidental rapid cycling of the O key, I also had to add a private member variable to track the state of the O key. Ensuring that the action only fired once per key press was essential to make cycling between orthographic views comfortable.

Overall, navigation was carefully designed not to only allow movement, but to give users the tools they need to explore the scene comfortably and reset back to a familiar position if they get lost, providing a comfortable experience.

**Custom Functions**

Throughout the project, several custom functions were needed to make the code for the scene manageable and more organized. Some of the most crucial functions I developed for this project were the render functions and input event callbacks.

Each render function, such as RenderCorkStopper(), RenderBook(), RenderChest(), RenderMug(), and RenderCandle(), is responsible for setting up the transformations, materials, and textures needed to draw each object. These functions include every necessary step to build each object from a combination of basic shapes like boxes, cylinders, planes, and spheres. By organizing these render functions in this way, the code was made more maintainable and readable. If the scene needed to expand into larger projects, these rendering functions could be re-used to speed up the development process.

I also implemented custom callback functions such as Mouse\_Position\_Callback() and MouseScrollCallback(). Mouse\_Position\_Callback() processes mouse events and maps them to movement of the cameras yaw and pitch inside the scene. MouseScrollCallback() changes the camera's movement speed based on scroll wheel input, allowing users to travel through the scene as fast as they consider comfortable. This design follows GLFW’s recommended use of input callbacks for efficient and responsive event handling (*GLFW: Input Guide*, n.d.). This allows responsive control over the camera without excessive polling for events.

DefineObjectMaterials() was used to load the material properties for each object into memory and based on a technique from LearnOpenGL’s Lighting Materials tutorial. By specifying ambient, diffuse, specular, and shininess values, I was able to simulate real surfaces like wood, metal, fabric, and tile, interacting with lighting in the scene (*LearnOpenGL - Materials*, n.d.). Following the suggestion to group material definitions in one function, it became quite easy to adjust how each different surface responds to light. Combining these materials with textures allowed me to introduce a level of depth and realism to the scene that would not be possible without them.

By thoughtfully selecting each object, designing intuitive navigation, and building reusable and organized functions, I was able to create a polished 3D scene that demonstrates both technical and creative skills. Each development choice, from materials to input handling, was made to enhance the user experience.

**References**

*LearnOpenGL - Materials*. (n.d.). <https://learnopengl.com/Lighting/Materials>

*GLFW: Input guide*. (n.d.). <https://www.glfw.org/docs/3.3/input_guide.html>