CS-330 Design Decision

Aydar Fayzullin

**Development Choices for the 3D Scene:**

I chose to create a coffee mug, a pen, a box, and a globe as the objects in my 3D scene. These objects are commonly found in the real world and provide a variety of shapes and complexity to work with. For the coffee mug, I decided to use a cylinder for the mug body because it closely resembles the shape of a typical coffee mug. The torus shape was used for the mug handle as it provides a simple and recognizable handle structure. By combining these two primitive shapes, I was able to create a representation of a coffee mug.

For the rectangular box, I used a cube as it’s a basic geometric shape, making it relatively simple to create and manipulate in a 3D environment. The 3D box closely resembles objects encountered in the real world, making it instantly recognizable to viewers. The straightforward nature of a 3D box allows for easy transformation. Scaling, rotation, and translation operations can be applied directly to the box's vertices, making it simple to create dynamic effects.

Similarly, for the pen, I chose a cylinder for the pen body as it represents the cylindrical shape of a pen. The cone shape was used for the pen tip, which is commonly seen on pens. This combination of shapes allowed me to create a simple representation of a pen. Lastly, I selected a sphere to represent the globe and a tapered cylinder for the globe base. The sphere accurately depicts the shape of a globe, while the tapered cylinder provides a stable base for the globe to rest on. These shapes helped me create a recognizable representation of a globe.

To achieve the required functionality, I utilized organized geometry to ensure that the polygons on each 3D model were well spaced and connected. I focused on keeping the polygon count below 1,000 triangles to minimize complexity and save modeling time. For texturing, I chose two objects to texture: the coffee mug and the globe. I applied accurately projected textures to these objects to enhance their visual appearance. By carefully mapping the textures onto the objects' surfaces, I ensured a realistic representation of the materials. In terms of lighting, I implemented a minimum of two light sources. I used the previously created light from a previous milestone and added another colored light source. I incorporated the components of the Phong shading model, including ambient, diffuse, and specular lighting, to create a polished visualization of the 3D models. To place the objects appropriately, I utilized the X, Y, and Z coordinates to position them relative to each other in the 3D world. I matched the selected photograph as closely as possible, ensuring that the objects were positioned correctly.

**User Navigation:**

To navigate the 3D scene, I implemented horizontal, vertical, and depth camera navigation. The camera can traverse the X, Y, and Z axes to capture all of the objects in the scene. The user can control the camera's forward, backward, left, and right motion using the WASD keys. The QE keys are used to control upward and downward movement.

For nuanced camera controls, the orientation of the camera can be changed using the mouse cursor. The user can look up, down, right, and left by moving the mouse. Additionally, the mouse scroll can be used to adjust the speed of camera movement, allowing the user to explore the scene at their preferred pace.

To switch between perspective and orthographic views, I implemented a keyboard key tap functionality. When the user taps the O key - the viewport display of all objects in the scene toggles between orthographic (2D) and perspective (3D) views - P key. The camera's orientation remains the same regardless of the chosen view.

**Custom Functions:**

I am uncertain if I have personally created any custom functions within my program. Most of the functions used were obtained from tutorials and instructional videos that I accessed throughout this class. However, most of these functions are modular and can be reused. The shader programs, camera position, and vertices can be easily altered. Despite the program containing a significant amount of code, it is quite straightforward to read and comprehend. The code is well-organized with appropriate spacing and comments, which facilitate understanding of the intended objectives. Modifying the vertices can be done without the need to rewrite an entire section of code, ensuring that the object will still render correctly.