Tiffany Gomez

CS-330-T5625

Dr. Diesch

Final Project

06//2022

**Design Decisions**

The 3D scene rendered in this final project is based on an image that consists of a table placemat, a napkin, a tea mug with a lemon slice, a plate, and a tomato. The placemat serves as the plane and the base of the scene. To render the plane object, a structure for the plane was created as well as a function to set up the buffers for the plane and a function to create a plane mesh. The plane mesh function includes the position, normal, and text coordinates that are stored in the plane structure. The function that sets up the plane buffers is responsible for activating the vertex array object for the plane shape, activating the first buffer, and creating the attribute pointers. As this was the second object that was built in the scene, it binds to the second vertex array object (1) and the third buffer (3). This same structure is adopted for the rest of the objects within the scene and is what allowed me to easily render other objects that share the same shape. For example, I was able to render the napkin object by simply changing the model view of the plane (adjusting the scale, rotation, and translation), binding the napkin to the vertex array set for the plane, and drawing the plane using the coordinates stored in the structure. This exact format helped me figure out how to create one of the most challenging aspects of my complex object (the mug), the handle. The original plan was to use a half-cylinder or torus, but both were at the moment beyond my level of skill, so instead, I followed the same structure of the plane to draw two cubes. At this point, I had already created my mug object and just used graphing tools to map the new model view for each cube that would place the handles on the mug.

As the scene’s complex object, the mug was designed to be built using, aside from the cubes for the handles, various cylinders to represent the mug, a cylinder for the tea, and a cylinder for the lemon. Thanks to the code I referenced from Song Ho Ahn’s website OpenGL, I was able to add a cylinder class to my program. This class helped me to realize I could modify the coordinates for the base and top to simplify my mug object to a single cylinder. Furthermore, Song Ho Ahn’s website provided a “WebGL Cylinder Interactive Demo” that let me visually experiment with coordinates that I was able to later input into three objects that reference the cylinder class for my mug, tea, and plate objects. I was able to duplicate this same format for creating a sphere class to reference the tomato object. I did simplify the creation of the mug, the plate, and the tomato, largely due to the skills I gained in adding textures and light sources. Instead of creating an additional lemon object to place in the tea, I instead chose to bind two textures to the tea, an image of the tea and an image of the lemon. Instead of using a large and smaller cylinder to create the plate, I instead chose a texture of a plate that had an additional drawn cylinder and used my lighting sources to play with color, ambient lighting, and specular intensity to specifically adjust their strength to create the illusion of depth and shine to the plate. Additionally, I was able to recreate this strategy for the tomato too, as I had planned on adding a smaller cylinder to represent the stem hole.

Early on in the construction of the scene, I incorporated the camera header from this course’s source code and modified it to include upward and downward movement, as well as a constant movement speed, using delta time, so that users can utilize the mouse and key movements to move through the scene’s world space. I referenced the camera class in the program that created a camera object with position, direction, yaw, and pitch coordinates that I was able to modify throughout the build process to best capture the scene. The ability to update the position of the camera in accordance with my input, allowed me to traverse my scene in six directions to adjust my object placement and light sources to illustrate as closely as possible the scene illustrated in my original image.

Incorporating functions, structures, and classes, as well as implementing best coding practices for formatting, and commenting on my program allowed me to develop reusable code that made my work more modular and organized. As the project progressed and I began to layer the objects of my scene, this structure became essential in allowing me to modify certain aspects of my program. For example, I created global variables for certain lighting components. This allowed me to modify, before rendering each object, the strength, color, and intensity for which each light source affects each individual object without having to create a light source for each object. Using structures to store the mesh data for each shape made it easy to reference the structure to render multiple objects. As such, creating classes for shapes and referencing the class with each object’s individual coordinates made it even more simple to create multiple objects of different positions, scales, and rotations while using the same shape. As with Object-Oriented Programming, all the elements in my scene, whether they be shapes, lighting sources, cameras, or user input, are stored within their separate functions and called upon for customization for each object in my scene during rendering. While I have a long way to go and believe I have a lot of room for improvement in terms of modularity and reusable code, I believe I have set a foundation that allows me to continue to experiment with my scene beyond this course.

**References**

Ahn, S. H. (n.d.). *OpenGL Cylinder, Prism & Pipe*. OpenGL. Retrieved June 13, 2022, from http://www.songho.ca/opengl/gl\_cylinder.html#webgl\_cylinder