7-1 Project

Tyten Perez

CS-330 Computational Graphics and Visualization

Ronald Bishop

February 25, 2024

As part of the planned scene from module 2, my project’s scene was a simple desk surface with a handful of small objects to re-create. These ranged from simple shapes to more complex objects requiring multiple shapes. I feel that these objects helped me understand the unique challenges and opportunities for creating, shading, and texturing 3D models. The first objects I set out to recreate were the Desk Surface (Plane) and the Puzzle Toy (Cube). This allowed for an approachable introduction into mesh creation and texture mapping, given their uniform and predictable surfaces. Then there was the AA Battery (Cylinder), introduced to tackle more challenging cylindrical geometry. In early iterations of this shape, I inefficiently calculated the vertices using manual mathematical equations. This proved difficult to maintain, especially when adding normal and texture data to the shape. To combat this, I utilized common coding techniques to generate the cylinder from formulas built into the UCreateMeshCylidner function. The last two shapes I implemented were the Toy Ball (sphere) and Phone Charging Adapter (several cubes), which used the same techniques of built in formulas and complex shape transformation respectively. Although I successfully recreated most of the objects in my 3D scene, I failed to complete the headphone ear-pad (Torus) in time due to its mesh complexity.

To navigate this scene using keyboard and mouse I utilized much of the camera class that was present in the GitHub tutorial projects. These included functionalities are using WASD and mouse movement to move and turn the camera. To support the additional elevation movement, I modified this class to allow for keystrokes to change the camera’s y-axis position with keys: Q or E. Additionally, to change the camera’s perspective, I used a bool variable in the URender() function to change from orthographic or perspective when the P key is pressed.

Some of the modular functions used in my project were the renderObject() and UCtreateMesh functions. These handle essential tasks like rendering objects and passing the geometric shape data. The renderObject() function handles the rendering process of each 3D object in the scene. By taking a mesh, a model matrix for position and orientation, a texture ID, and a location for the model matrix uniform, it draws an object with the correct transformations and applied textures. The UCreateMesh functions like the UCreateMeshCube, were developed to generate the basic shapes required for the scene's objects. These are then translated and resized as needed across the scene in the URender() function. For example, UCreateMeshCube is called multiple times to create both the puzzle toy and the components of the phone charging adapter objects. These functions simplified the rendering and shape creation processes as it reduces redundant code in the project.