Marye Bierbaum

CS-330 Comp Graphic and Visualization

Final Project Reflection

For my final project I reconstructed an image I’m calling ‘Game Night’. The 2D image featured various items from games, such as dice, meeple, and marbles, with a boardgame box and a dice tower in the background. I chose these items because they are immediately recognizable as game artifacts and also because of the simplicity in their shapes. Dice are just cubes, marbles are spheres, and the boardgame is a large rectangle. To start laying out the scene I began with my white table surface and the blue wall background. Then I positioned the largest and simplest object, the boardgame box, on the corner of the table and found the middle of the scene. That became my guide for placing the remaining shapes.

The meeple and dice tower, which are the complex items in my scene, can be broken down into smaller, less complicated shapes as well. Each of the meeple start with a pyramid body, are topped with a sphere head, and completed with two mirrored rectangles for arms. Once I built and appropriately scaled one meeple, it was easy to generate the additional three. I just had to position each of their components to the same location and make sure their rotations were equal. The dice tower features two rectangles: one as the base and one as the dice drop. They are separated vertically by a few points of space to account for where the dice come out of the tower. Both rectangles are encased by three slabs, making the right side, left side, and back of the tower. Because the tower is sitting at an angle in the photograph, lining up these pieces was the most difficult of all the positioning in the 3D scene.

To recreate the lighting in the scene I used two different light sources: one is a white light with 80% specular intensity and one is a blue colored light at 30% specular intensity. The two lights sit ten points away from each other x-axis, but share y-axis and z-axis coordinates. I chose these two lights and their positions because they best matched the cool light source in the original photo.

To navigate my scene, the user will need a keyboard and a mouse. The keys W, S, A, and D will move the camera location forward, backward, left, and right, while the Q and E keys will move the camera up and down. Moving the mouse will change the orientation of the camera, or the direction it points. Finally, scrolling the mouse scroll wheel while moving the camera will change the speed at which the camera moves.

Throughout the code I utilized functions to make my code more modular and organized. The function that renders my objects (URender()) houses the information for the light sources and the shapes in the scene. Here we determine where the objects will exist, its size and rotation, and is texture. Each of those properties are established outside of URender(), so they just have to be set within the method. This makes reading and manipulating those properties much more straight forward. Additionally, anything that needs to be defined or loaded is done outside the URender() method as well. Textures are defined in the UCreateTexture() method, shaders in the UCreateShaderProgram. This makes the code very modular and very easy to pull in and adjust to get the correct shapes and lightening.