**CS 330 Project Reflection**

Cydnie Fisher

Southern New Hampshire University

CS-330 Comp Graphic and Visualization

Professor Phil Enkema

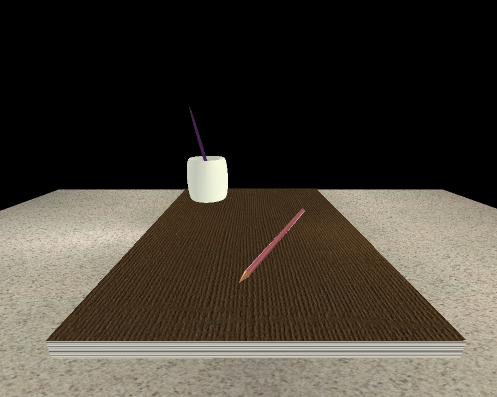
December 15, 2024

**Development Choices**

When choosing which scene, I wanted to recreate for the final project, I wanted a scene that was dynamic in regards to shapes, textures, and light reflections. This scene consists of two different complex shapes: a cup where the sides are a torus and the base is a cylinder, and a pencil where the tip is a cone, and the base is a cylinder. It also consists of a paintbrush made from a tapered cylinder that is located inside the cup, which adds more complexity to the scene. Finally, it consists of a drawing pad made from a rectangle and a counter that is made from a plane. This scene consists of 5 items, with 6 unique shapes being used.

I also wanted to ensure that the objects consist of many different textures which react to light differently. The cup’s texture was simplified by only having a solid color which reflects a lot of light, resulting in stark contrast between the light and shadow. The base of the pencil and the paintbrush are composed of one color and share a similar shine which results in a bright highlight where the light hits. The tip of the pencil is a different texture from the base of the pencil and reflects less light. The drawing pad consists of two different textures, one for the cover on the top and bottom and one for the paper on all sides. Both of the textures absorb most of the light, resulting in less of a highlight. Lastly, the counter consists of a speckled texture which also reflects a lot of light.

**Original Photo Recreated Photo**

A magazine with a cup and pencils on it

Description automatically generated

**Navigation**

The user is able to navigate the 3D scene through two different input devices: the mouse and the keyboard. Moving the mouse around on the screen allows the user to change the orientation of the camera without changing the camera's location. The scroll wheel can be used to adjust the speed of movement and the speed that the camera can travel. Scrolling upwards causes the speed to decrease while scrolling downwards causes the speed to increase. Mouse events are handled by the Mouse\_Position\_Callback and Mouse\_Scrollwheel\_Callback functions within the ViewManager class. If the user wants to change the camera's location, they can do so using the keyboard. The WASD keys can be used to move the camera forward, left, backward, and right respectively. The QE keys can be used to move the camera up and down respectively. The keys OP can be used to switch between orthographic (2D) and perspective (3D) views while maintaining the orientation of the camera. Lastly, the ESC key can be used to exit the program. Keyboard events are handled by the ProcessKeyboardEvents in the ViewManager class.

**Custom Functions**

Overall, this code is well organized and consists of many functions that are called upon multiple times within the code. These implementations make the code easier to read, maintain, and write and helps to avoid redundant code. The SceneManager and ViewManager classes were worked on by me. The SceneManager class is responsible for managing the preparation and rendering of the 3D scene and includes code for the 3d objects, such as the shapes, textures, and lighting used. The ViewManager class manages the viewing of the 3D objects such as camera and projection angles.

In the SceneManager class, the CreateGLTexture method loads textures from images, maps those textures, and loads the textures so they can be read. This is called upon any time a texture is added to an object. The SetTransformations method is used to pass down transformation values. This is called upon for all shapes to ensure their scale, rotation, and translation. The SetShaderColor and SetShaderTexture methods set color values and bind textures respectively. The SetShaderColor method is called on when a shape can be a solid color, while the SetShaderTexture is called on when an image needs to be set as a texture for a shape. The DefineObjectMaterials method defines the diffuse and specular color, shininess, and tag of each object material. This can then be called using the respective tag to determine how each shape reacts to light. The SetupSceneLights configures the lighting sources for the scene, such as their lighting type, direction, ambiance, diffuse and specular values. The PrepareScene method prepares the 3D scene, loading the textures, object materials, lights, and shapes. Lastly, the RenderScene method renders the scene by rendering shapes, applying textures and materials, transformation values, and configuring the lights.

In the ViewManager class, the CreateWindowDisplay function creates the main display window, handling GLFW configurations such as mouse and scroll callbacks. The Mouse\_Position\_Callback and Mouse\_Scroll\_Wheel\_Callback functions process mouse movement and scroll events which adjust the camera’s view and zoom respectively. These events are called on when the mouse cursor moves, or when the scroll wheel is used. The ProcessKeyboardEvents method processes keyboard events and is called upon when any key is pressed. Lastly, the PrepareSceneView method prepares the scene by processing keyboard events, camera view, and projection.