Kileigh Adams

SNHU

CS 330

October 19, 2025

Final Project Reflection

For my 3D scene, I recreated a café table inspired by one of my favorite cafés in New York City. The warm and relaxed feel of that café, with its natural wood textures and soft lighting, made it the perfect environment to build in 3D. My goal was to create something small but believable, using the provided OpenGL framework and expanding it with my own models, textures, and lighting choices.

The main objects in my scene are a wooden table, a ceramic coffee cup and saucer, a leather notebook, and a metal pen. I chose each shape with a purpose; I selected the mug because it could be recreated using a modified cylinder mesh, which made it both challenging and manageable for the skills I had at the time. The saucer and table were built from simple flattened cylinders and planes, which helped me understand how scaling affects geometry in space. The notebook began as a box mesh with scaled dimensions to mimic a thin hardcover, and the pen was created from a stretched cylinder to test lighting on a reflective surface. These simple forms allowed me to focus on materials and light behavior without getting lost in overly complex modeling.

Throughout the milestones, I ran into several challenges that helped me grow as a programmer and designer. At first, my textures appeared stretched or misaligned, which pushed me to experiment with UV scaling and seamless textures. Later, my lighting looked flat, which led me to discover that some vertex normals were missing. Fixing that problem was a breakthrough moment because it helped me understand how light interacts with surface orientation. I also had to adjust the pen’s rotation to face the correct direction, which taught me the importance of transformation order. Toward the end, I added a high-resolution café interior image as a background to make the scene feel more realistic and visually connected to its environment. I used transformation functions to position, rotate, and scale each object in world space so that they appeared natural in proportion to one another.

For texture mapping, I worked with the SetShaderTexture() and SetTextureUVScale() functions, which used the stb\_image library to load image files and generate mipmaps. These functions ensured that textures displayed correctly and stayed sharp when viewed from different distances. The SetShaderMaterial() function let me control how each object reacted to light. For example, the wooden table had a low specular value to reflect light softly, while the pen had a higher shininess to simulate a metallic surface.

While working with the lighting system, I spent a lot of time tweaking small values for object positions, light colors, and intensity levels. Experimenting this way gave me a much clearer sense of how objects exist in three-dimensional space and how subtle changes affect realism. That practice also gave me an entirely new appreciation for the complexity of 3D animation. I now understand how much invisible work goes into crafting believable environments in animated movies, and I will never look at films like *Shrek* the same way again.

I expanded on the existing camera controls to allow for natural exploration of the scene. The camera was implemented using glm::lookAt, which calculates the view matrix from position and direction. The W, A, S, and D keys move the camera horizontally, Q and E move it vertically, and the mouse adjusts yaw and pitch for rotation. I also added zoom functionality using the scroll wheel, which modifies the field of view and lets users focus on individual objects such as the mug or pen.

At one point, the camera speed varied depending on frame rate, which made movement unpredictable. I fixed this by multiplying motion by deltaTime, ensuring smooth and consistent navigation. These controls make it possible to view the café table from different perspectives, much like how someone might look around a real table in person.

To keep the program organized and efficient, I built on several reusable functions from the project outline. The SetTransformations(scale, rx, ry, rz, pos) function calculates the model matrix and applies transformations for each object. This kept my workflow consistent when placing new meshes in the scene. The SetShaderMaterial(name) and SetShaderTexture(name) functions manage how surfaces interact with lighting and texture properties, while SetTextureUVScale(u, v) controls how textures tile on a surface. These reusable tools made it easier to fine-tune specific elements without touching the core rendering logic. Each object has its own draw function, such as DrawTable() or DrawPen(), which calls the transformation, texture, and material setup in order. This modular approach made the code easier to troubleshoot and expand.

I made my code more modular by separating each responsibility into small, reusable functions that handle one task at a time. Instead of hard-coding transformations or lighting settings for each object, I created parameter-based functions that could be reused across multiple draw routines. This meant I could easily add new objects, modify material properties, or adjust lighting without rewriting existing code. It also made debugging more manageable because I could isolate problems within individual functions.

Building this café table scene with the provided OpenGL framework helped me understand how to turn structured starter code into a personalized, functional project. Each milestone added complexity, from object creation to texturing and lighting. Experimenting with transformations and light values gave me a deeper understanding of how realism is created in three dimensions and a much stronger respect for the artistry behind professional animation. By the end, I felt more confident not only in my technical skills but also in my ability to bring a real place to life through code and design.

**References**

de Vries, J. (2014). *Lighting: Colors.* LearnOpenGL. [https://learnopengl.com/Lighting/Colors](https://learnopengl.com/Lighting/Colors?utm_source=chatgpt.com)