7-1 Final Project Reflection

David S. Harmor

Southern New Hampshire University

CS-330

Professor Aly

February 20, 2025

Not being artistically inclined I was a bit intimidated by this course. When it came time to select a scene image to render, I looked ahead as to what we had to do with the assignment. I found that the assignment called for duplicating the scene with basic shapes. I also looked at the 1-2\_OpenGLSample project provided to us in the course and examined how the objects were created using just basic shapes. Specifically, I examined the wine glass since it was comprised of multiple shapes and how the textures and lighting were done. Using this information, I looked at multiple images of objects, I found a pen and a coffee mug image and knew I wanted a scene that included those objects. So, I searched and found an office-type setting that included those objects and based on the information I gathered I had a chance of creating a 3D rendering of it. To create a close approximation of the textures of the objects I used Adobe Photoshop to capture the colors and take a sample image of the object for the texture. Milestone One of the project was very helpful, it made me think of how the objects can be created in the rendered image using basic shapes. The framework provided has its good points and bad points. The good points are that it gave a level of abstraction of the OpenGL functions and allowed me to concentrate on the project and images. That high level of abstraction also prevented me from creating objects in a manner I thought was needed. For instance, I had a jpeg image of a window that I wanted to include so I looked on the internet to see how to accomplish that I found a site that described how to do so but it involved low-level OpenGL functions which seemed to be out of scope for this class. and it wasn’t clear how to incorporate the code with the framework. So a technique I used was to create a box with the image of the window as the texture and I made the thickness of the box very thin. I initially tried using a plane for the window image, but it was coming in about 45 degrees counterclockwise. When I changed it to a box I had the same issue, after trying multiple methods I found that the jpeg image was the issue, any attempt at modifying the image in Adobe Photoshop or Gimp caused the image to be skewed. After trying different methods I finally created an image that came in correctly. I wanted to create a bevel so the window looked “sunk” into the wall but any attempts at modifying the image would cause an issue when applying it to an object. The solution I used for the window, creating a box and adding the image as a texture, I was able to use it in creating the image of the laptop. Once I was able to create those objects the rest just fell in line as to how to create them. Although it’s not a 100% match to the original, I feel the spirit of the image was kept.

|  |  |
| --- | --- |
| Original | Rendition |
|  |  |

The navigation component of the application contains a variety of functions that the user can use to manipulate what is shown on-screen. The standard motion of the camera uses the WASD keys to control the forward, backward, left, and right motion, Q and E for up and down movement, and the mouse to change the view of the camera. These keys/motions use the ViewManager predefined motions to adjust the camera’s position within the scene. Additionally, the keys O and P change the view to Orthographic Projection and Perspective Projection, respectively.

For each of the objects I created a custom function that is designed to be reusable, it takes the position, scale, and rotation as parameters. By modifying the parameters passed into a method for creating an object, the object can be reused by changing those parameters and calling the method again. This also reduces the code sections and makes it easier to find each object called since the code for creating the object is kept separate and if the object is a compound object the components are created relative to the main object. For example, the coffee mug is comprised of a cylinder for the base mug and a half torus for the handle. The handle is created relative to the position of the mug/cylinder. This allows the mug object to be relocated and has the other components that make up the object stay in place relative to the main object. As a result to create another copy of the object or to relocate it the calling function parameters just need to be updated. For the most part, this worked out well except for the pen, I created 3 instances of the pen in different angles 1 calling the initial pen creation which lies on a pad at a 45-degree angle, and the other two are placed vertically inside the pen holder. The pen is comprised of a cylinder and a tapered cylinder, I ran into an issue with orienting the pen tip (tapered cylinder) with the base of the pen. I was unable to get the angle to be relative, so I created a second pen method due to the time constraints.

One issue I had with the frame worked which caused me to lose a lot of time was the textures. When I added a texture to the list for creating textures the program crashed on startup. Since I had just created the image, my first thought was the image was corrupt. So I did a lot of testing on the image to ensure it worked. After some time I noticed a comment in that code section at the bottom stating only 16 textures are allowed and searching the internet I found this to be a limitation of the OpenGL library. So, to fix this issue I added the following code to the function call:

bool SceneManager::CreateGLTexture(const char\* filename, std::string tag)

{

….

int Max\_Textures = 16;

// Test if texture slots are full

if (m\_loadedTextures >= Max\_Textures)

{

std::cout << "Texture slots are full. Total count : " << m\_loadedTextures<< std::endl;

std::cout << "Could not load image:" << filename << std::endl;

return false;

}

…..

}

This code allows the program to load and run if more than 16 textures are defined. It will fail to load these textures and display a message in the console window as to why. Even though I noticed the comment at some point in regards to the 16-texture limit I didn’t realize I had exceeded that limit without having to physically count all the textures.