Mac Grier

CS 330

8/17/2025

**Final Project Reflection**

For my scene I chose some objects on the counter in my kitchen. I chose to model the toaster, the plate it sits on, the candle, the lighter, and the butter dish. I ignored the cords and some minor details like the wall socket because they would be either hard to represent, or I deemed them less worthy of development time.

To create my scene, first I created the countertop and the wall behind it. Making these planes first created a platform for my objects to be placed on. Then I started creating my objects. The easiest object to create was the candle which simply used two cylinders with one as the base and the other as the slightly larger lid. The hardest object by far was the toaster. If I had known how much trouble it would be I might have done a different scene! It ended up being a complicated series of boxes used to create the empty space required for the pusher and the toast slots.

I added camera controls to help viewers navigate the 3D scene. These controls are:

* W – Moves the camera in
* A – Moves the camera to the left
* S – Moves the camera out
* D – Moves the camera to the right
* Q – Moves the camera up
* E – Moves the camera down
* O – Changes the perspective to an orthographic view
* P – Changes to the normal perspective
* Scroll Wheel – Slow or quicken camera movement speed

To help me create this scene I created the function LoadSceneTextures(). This function took the textures in my program’s folder, made them into GLTextures, and bound them for use in the scene. Without this function my objects could only have been colored, and the scene would have lacked detail.

Another function I created was DefineObjectMaterials(). In this function I created object materials that I could apply the meshes I created. These materials had a shininess value as well as values that dictated how light would be affected when it landed on them with diffuseColor and specularColor parameters. Applying these to my scene made it feel much more real because my objects interacted with the light.

A function that I didn’t create but added a lot to is ProcessKeyboardEvents() in ViewManager.cpp. Here the program processed input from users and turned it into actions on the screen. This is where camera movement and perspective shifting keys were added. This function is expandable and reusable in other programs because it has a simple form that can be easily added to. If any more features are needed for the camera, like more perspective shifts, this is where they will be added.