Kenneth Dandrow 2/22/25

Prof. Brian Holbert CS-330

**Module 7: Final Project Reflection**

****

**Choosing Objects and Placing Them in the Scene**

For this project, I created a 3D scene of a desk using OpenGL. The scene includes several objects you would normally find on a desk, such as a lamp, a computer monitor, a keyboard, a mouse, a book, a cup with pencils, and even a donut. These objects were made using basic 3D shapes like boxes, cylinders, and a torus (for the donut).

Textures were applied to some objects to make them look more realistic. For example, I used a wood texture for the desk and a screen image for the monitor. Some objects, like the lamp and book, use materials instead of textures, which allow them to have a solid color with realistic lighting effects.

One of the biggest challenges was making sure textures were applied correctly. At first, when we added a texture to the book, it also affected other objects. This happened because the shader program wasn’t resetting the texture state after each object. I fixed this by making sure each object correctly sets its material or texture before rendering.

I carefully placed the objects in the scene to make it feel realistic. The lamp was moved to the back-left corner of the desk, with the cup of pencils placed in front of it. The keyboard and mouse were positioned in front of the monitor, and the book was rotated slightly to make it look more natural. Finally, I placed the donut near the cup as a fun little extra detail.

**Navigating the 3D Scene**

To allow users to explore the scene, we created a free-moving camera that lets them look around. The controls are similar to those in first-person video games:

* WASD keys move the camera forward, backward, left, and right.
* Q and E keys move the camera up and down.
* Mouse movement changes the direction the camera is looking.
* The scroll wheel adjusts the movement speed, making it easier to navigate.

Another important feature I added is the ability to switch between different camera views. By pressing the "P" key, the user can toggle between a perspective view (3D) and an orthographic view (2D-like). This was done by switching between different projection matrices in the ViewManager class.

**Adding Lighting to the Scene**

Lighting was a key part of making the 3D scene look realistic and visually appealing. I used three main light sources: two desk lights positioned on the left and right, and a soft overhead light acting as ambient room lighting. Each light was carefully placed and angled to evenly illuminate the objects while preventing harsh shadows. The desk lights provide direct illumination with a soft spread, while the overhead light ensures that the scene does not appear too dark. To enhance realism, we adjusted the ambient, diffuse, and specular components of the lighting, which control how objects reflect light and create highlights. I also fine-tuned the specular intensity to prevent overly shiny surfaces and adjusted the focal strength of each light to ensure a natural distribution. These lighting adjustments brought depth and dimension to the scene, making the objects appear more lifelike and improving the overall visual quality.

**Making the Code Modular and Reusable**

To keep the code organized and easy to update, separate classes were used for different parts of the program.

The SceneManager class handles creating and positioning objects. It includes functions like SetTransformations(), which allows us to scale, rotate, and move objects easily without having to rewrite a lot of code.

The ViewManager class is responsible for camera movement and user controls. It includes functions like ProcessKeyboardEvents(), which handles WASD and other key inputs, and Mouse\_Position\_Callback(), which lets the camera turn when the mouse moves.

The Camera class contains functions like ProcessMouseScroll(), which adjusts movement speed, and GetViewMatrix(), which updates the camera’s position and direction.

Another function created was called SetShaderMaterial(). It applies materials to objects. Instead of setting colors and lighting properties manually for each object, we can just call this function with a material name and it will apply the right settings automatically.

By organizing the code this way, the project is easy to modify and expand. If we want to add more objects, we can reuse these functions instead of writing new code from scratch.

**Conclusion**

I had a lot of fun working on this project and learned a lot about 3D graphics, textures, lighting, and camera movement. At first, it was a bit challenging to understand how transformations, shaders, and materials worked, but after spending time troubleshooting and experimenting, I started to understand how everything fits together.

One of my favorite parts was adding textures and making sure everything looked realistic. I also enjoyed figuring out how to make the camera movement smooth and easy to use, since that made exploring the scene much more enjoyable.

Overall, this project was a great experience, and I feel like I now have a better understanding of OpenGL and 3D rendering. I would love to work on more projects like this in the future and expand on what I learned in this class.