Tarik Iles

CS-330 OpenGL

Southern New Hampshire University

7-10-2024

Final Project Reflection and Design Decisions

A screenshot of a computer program

Description automatically generatedThroughout my curated 3D scene, I constructed a crude representation of my cat, the Luka-man, and his surrounding environment, a chair he sits upon, a vent behind the chair, a fridge to his left, and a doorframe to his right. These objects tend to be simple, apart from the complex chair and more complex cat. I chose them because of the escalation of model complexity, the variety in materials, the placement of the room light and window, and because while it’s not overbearing and its objects are limited producing this would be conducive to my growth as a graphics programmer. To create the objects, I created a series of local-model-space modifiers for each complex object grouping, i.e. the position of the chair or the cat, and their angles. Admittedly, the local space for the angles wasn’t perfect because the rotations did not maintain relative space to the models, but the local spatial modifier for X, Y, and Z coordinates worked fantastically for moving the entire chair or cat around as needed.

I also included two lighting sources; From behind the hind wall and the doorframe, I included a soft yellow light from the sun, and from the center of the room I had a dim white light for the LED bulb shining from the ceiling. The entire project revolved around following repetitive patterns. In order to define materials, I needed to set ambient light colors and strength, diffuse color, specular color, and shininess. Afterwards, I could define a tag and push\_back the material into the objectMaterials list. I did this for each of the most generic, general material types in my project; Drywall, steel, wood, and fur. I also included two materials for my outlet and vent respectively. To add textures, I had to import them to OpenGL with the CreateGLTexture command, which only worked on variations of the 512...1024…2048...4196 sizing scheme. This did involve me creating custom textures by having my father send me photos of my cat from other angles, from which I modified the files into textures to wrap onto the primitives which composed him. When drawing the primitives I had to first load the general mesh into the scene preparation Init, then before drawing set transform parameters, i.e. scale, rotation, and position. If the object was appropriately complex, I would add a grouping transformation variable set so that I could move all of the object’s components (chair, then cat body, then cat head) together without upsetting their position relative to the other components. After, a shaderColor or a shaderTexture could be assigned as well as an appropriate shaderMaterial. If a shadertexture was chosen I could set the UV scale to suite my needs, generally being 1.0f, 1.0f to stretch one time vertically and one time horizontally. For the fridge I made the texture stretch many times more vertically than horizontally and applied a tiled texture wrap. I also discovered that the objects were drawn in the order in which the script called them to appear, so I could skip scaling on individual objects that were congruent, i.e. the cat’s arms and ears, making finer adjustments easier to integrate. An issue I ran into as a limitation was that I could not find a good way to not render portions of a primitive model; Consequently, to hide the torus that made up the back of the chair, I simply hid the bottom of the chair with a black plane. I accepted the cat’s ears being complete cones instead of half cones because I found that it still looked generally good.

To navigate this scene, there are WASD controls for moving forward, left, in reverse, and to the right. There are also Q and E controls to lower and raise the camera. I attempted to add scrolling speed control, and discovered there was already a clamped minimum and maximum speed, but in practice I ran into an issue with a “GLFWScrollFun” datatype desiring to be present in the stead of the GLFW window instance datatype, meaning that I was unable to implement it. Finally, O and P respectively toggle the orthographic or perspective displays of the 3D environment I created, allowing the viewport to be changed b the end user dynamically. In the ViewManager script I had all of the keybinds following patterns that utilized if statements and GLFW keypresses to apply a function to move the camera in accordance with the time pressed and the camera’s movement speed. Artificially adjusting the movement speed during testing proved the movement speed worked in its implementation, apart from the scrolling, which I could not find a way to implement. Other patterns I saw to were calling functions and subroutines every frame through the main function, such as calling ProcessKeyboardEvents in the PrepareSceneView every frame, along with a camera ViewMatrix and ProjectionMatrix. I believe everything I’ve done was applicable to other OpenGL based applications that could be programmed because of how similar these patterns were. I did try to automate the creation of the chair backing rods, which would have been a function utilizing a for loop to create eight rods and alter their height to form a parabellum, though this only led to all of the legs drawing into each other, despite my attempts to keep them separate by creating individual new cylinders each cycle of the for loop. This resulting likeness was my final output. It is the best 3D model of my cat A cat sitting in a chair

Description automatically generatedever rendered.