Student Information

Name: Sofia Beyerlein
Cornell NetID: sb2669
GitHub Username: sofia2504

Unit Test Results

Score (out of 0): 0
Score %: 0
Total Tests: 0
Total Passed: 0
Total Failed: 0
Total Skipped: 0
Total Inconclusive: 0

Links

GitHub Repository Link:
`https://github.com/cs5650-2024fa/h-meshes-sofia2504/tree/student`

Video Recording Link:
`https://drive.google.com/file/d/1OGwtFye0-zbjTgrx6gwLLnvp8m_xu3gd/view?usp=drive_link`

Quest APK Link:
`fieldnotfound`

Work Summary

In this assignment, we had to render different objects using different techniques from ch.7. We used lam- bertian shading, blinn-phong, rasterization, and barycentric coordinates. My initial approach was to start from the first method and work my way down. As I worked on the methods, I would reread the section of the textbook that spoke about that particular method. When it came to the formulas, I tried to break them up into as many components as possible in order to not mess up any of the order of operations. As I started implementing the first few functions, the green cube rendered and I believe the cylinder in the back rendered as well. However, I did run into numerous issues that I worked on throughout the rest of the week. The first error was that I did not multiply the ray direction by negative one at first. I realized a few days after, when I looked into the bottom right (screen geometry) and remembered what the professor said about the viewing plane being on the negative z axis. I did not work on CalculateNormal() because I did not know where to start, so instead I moved on to the other functions. I made a mistake in CalculateLightDirection() where I started by doing hitPoint - lightPosition, but after I reread the textbook I realized my mistake and was a pretty easy and quick fix. It took me a few trials to do CalculateMappedNormal because I did not understand barycentric coordinates but after some supplemental youtube videos in addition to the textbook I realized it was not bad at all and also resulted in an easy fix. Lastly, CalculateNormal() was the last function I implemented. What ultimitaley made me resolve the function is by googling the RaycastHit implementation on Unity. I started by writing down the code verbatim and changing their vertices into the vertexNormal1, vertextNormal2, vertextNormal3 and set the barycentric coordiante equal to hitTransform.Transform(barycenter). I set the normal to the vertex components multiplied by their respective barycentric coordinate and added the terms together. This made the left pill look red and it seemed a little smoother than the right pill still not quite right. I then made the change of setting the barycenter to hit.barycenter since I realized this was an option while coding it on ryder. After much trial and error I then read the documentation of meshes and tried to do mesh.normals instead of mesh.vertices because I read that mesh.normals is used for calculating light and mesh.vertices was used for shape. Since the color was off, I figured that was the error and it is ultimately was fixed the left pill and red cube.

Test Suites:
Individual Test Results: