Project Checkpoint

Progress Update

Currently we have successfully implemented a method to parse an `.obj` file and store the mesh of triangles into the GPU memory. After the triangles are copied to the device memory, we are able to correctly render the object using ray casting to determine the correct triangle intersection and interpolate the vertex normal to compute the normal map of each pixel being rendered. Furthermore, we have implemented a basic "lambertian" shading model to compute the color of the intersected triangle point based on the light source position.

This matches the timeline of our project plan which also gave us extra time to read up on literature and useful libraries that will be paramount for the next steps of our project.

Implementation Details

Implementation:

-

Performance Measurements: Runtime Measurements:

- 960 triangle mesh runtime: 13.685 ms
- 4753 triangle mesh runtime: 66.592 ms
- 18192 triangle mesh runtime: 248.574 ms

NCU Measurements:

- SOL Compute (SM) Throughput: 96.41%
- Achieved Occupancy: 96.20%

Project Plan Revisions

Clearer Objectives have been set for the next steps of our project:

- Cornell Box Scene
- Monte Carlo Path tracing
- True Lambertian Diffuse Reflection