Project Proposal: GPU accelerated ray tracing

William Zhang and Sasha Makarovskiy

April 15, 2021

1 Objective

By the end of this project, we should have a working ray tracer capable of generating scenes in real time.

2 Requirements

1. GPU: We own one in my own machine which will allow me to see whether our product is working

3 Summary

Ray tracing involves casting rays corresponding to each pixel on the screen into the world and coloring the pixel dependent on the objects that it intersects. As each pixel is independent of one another, ray tracing is highly parallelizable. However, doing so on the GPU does pose problems due to secondary rays that are constructed during reflection and refraction. In this project, we will learn how to deal with these challenges and create a GPU-accelerated renderer.

4 Timeline

Below is our proposed timeline for this project. Italicized bullet points indicate topics pertaining to our graphics final project

1. **Wednesday, April 21:** Implement the ray tracer with support of
   (a) Displaying images on the GPU to the monitor
   (b) Triangle mesh intersection
   (c) Phong illumination model
2. **Sunday, April 24:** Improve on the ray tracer
   (a) Secondary ray calculation
   (b) Bounding volume heirarchy tree for pruning intersection search space
   (c) *Texture mapping of surfaces*
3. **Wednesday, April 28:** Implement a scene

(a) *Procedural generation of a world*

(b) *World, camera, and local coordinate spaces*

(c) *GUI and user input and output*

4. **Monday, May 1:** Real-time optimizations and presentation preparation

5 Measurements

1. The primary measurement we will be taking is the frame rate of the program. Should our program reach a framerate greater than or equal to 60, we would consider the program to be functional in real time. We anticipate that simple GPU acceleration would not suffice to reach this mark, and substantial optimizations involving pruning objects from intersection is needed.