COSC4370 Spring 2020 HW5 - Ray Tracing

April 13, 2020
Due: April 27, 11:59 PM, 2020

1 Introduction

In this homework, you will be adding some features to a barebones ray tracer using techniques we have discussed in lecture. Unlike our other assignments, we will be leaving nearly all of the software engineering, implementation, and structuring decisions to you. Start earlier than previous homework!

The ray tracer provided in hw5.zip provides only the simplest functionality. Be sure to read through the following files before beginning the assignment:

- **Image.h/cpp**: Provides simple functionality for writing .png images; requires libpng to compile.

- **Scene.h/cpp**: Stores the objects in a scene and performs basic ray tracing. Shape.h/cpp: An abstract class for representing a shape; also defines Ray and HitRecord structs.

- **Sphere.h/cpp**: An example child class of Shape.

- **main.cpp**: The entry point for the program. Sets up a scene, initiates ray tracing, and saves the output.

2 Setup

For convenience, we have included the Eigen library for linear algebra. You can see samples of the library in use for processing three-dimensional vectors within the code; visit eigen.tuxfamily.org for full documentation. Eigen is used in many important libraries, so it's worth exploring a bit.

As you work on this project, you will be adding several classes that do not currently exist. You should add appropriate files in the src/ and include/ directories and modify the Makefile accordingly. The starter program has been tested on Windows 10 with VisualStudio 2017.

3 The Main Assignment

Problem 1 (25 points). The ray tracer currently only can draw spheres. Add at least one more shapes. Potential options include other primitives, smooth surfaces with closed form equations $f(x) = 0$, etc.; remember that all you need to add a shape to your ray tracer are methods for ray intersection and computing normals.

Problem 2 (25 points). Right now our output looks pretty boring since there is no color and all the textures are Lambertian. Add support for Phong shading as discussed in the last assignment, and make sure each shape can have its own unique colored material.

Problem 3 (25 points). Add support for casting shadows.

Problem 4 (25 points). Add support for mirrored surfaces that reflect rays of light according to the law of reflection discussed in class. Be sure to limit recursion depth so that your ray tracer does not get stuck.

4   Deliverables

Submit all deliverables to your Github reporsitory.

- all the source code

- a screen shot of results for each of the problem.