COSC342 Assignment 2a

Due: 11:59pm, Tuesday 19th May 2020

Building a ray tracer

For this assignment you are asked to write parts of a ray tracer. Starting code for this assignment is available on Blackboard. **You must use the skeleton code and only modify the clearly indicated sections of the project**. We will mark your assignment against a reference implementation, so that we can independently test different elements of your code.

1. Starting with the skeleton program that we have provided, write and test routines that ray trace and shade the objects in the scene, meeting the requirements below. The skeleton program defines the required user interface, reads scene information from a file, and sets up many useful data structures. The format of the scene files and the functions and methods provided in the skeleton code are documented online.
2. Test your program with our sample files (text files in the skeleton code) and with scenes of your own design. Choosing appropriate test cases is part of the assignment.
3. Write a short report, describing how your program works and how you tested it. Describe any known flaws in your program.
4. Submit through Blackboard:
   - The *source code* for your solution to the assignment.
   - A sample image file and the scene description file to generate it.
   - A short report as a PDF file. (This should usually be up to two pages in length, and should not be more than three pages long.)

   We only need the files you have modified, as we will be testing them in that framework. Please DO NOT Submit:

   - Build directories, executables, or intermediate object files
   - Source control files or operating system generated files (e.g. `.DS_Store` from OS X)

Requirements for your ray tracer

We require that you complete the following tasks within the structure of the skeleton code provided. Some of these will be covered in labs.

1. The skeleton provides only ambient lighting. You should add Lambertian (diffuse) and Phong (specular) illumination.

2. Implement shadows and mirror reflections.

3. Implement rendering for `Plane`, `Cube`, `Cylinder`, and `Octahedron` objects.

4. Implement illumination from `DirectionalLightSource`s and `SpotLightSource`s.

See the code documentation for more detail on the required components.

Note that the scene file parser already recognises syntax relevant to the above tasks. The skeleton code is designed to provide you with implementation hints, and to allow you to focus on the graphics implementation required. See the To Do list in the documentation for information about which methods you need to update.

Marking scheme

4 marks	Report, with an emphasis on testing.
2 marks	Your sample scene image and associated scene description file.
1 marks	Diffuse (Lambertian) shading.
1 marks	Specular (Phong) shading.
2 marks	Shadows correctly cast by objects.
2 marks	Mirror reflections correctly computed.
1 mark	Correct rendering of planes.
1 mark	Correct rendering of cubes.
2 marks	Correct rendering of cylinders.
2 marks	Correct rendering of octahedra.
1 mark	Correct lighting from spotlights.
1 mark	Correct lighting from directional light sources.

Note that many of the components are independent. For example, you could complete diffuse shading, shadows, and cube rendering without cylinders, specular reflection, or mirror reflections. However, it may be difficult to test if your object normals are correct with just ambient lighting.

Final notes

Remember this is an assignment in graphics, not in production programming. You will not get any extra credit for extravagant solutions. High marks will be given for a well explained solution that is easy to follow. Programs must be commented of course, but no more than necessary for us to be able to understand them.

You may discuss conceptual issues relating to this assignment with others, but all the work you hand in must be your own, except for the parts of the given skeleton program.