CSC305 Assignment Four
The Glorious Ray Tracer

First Sphere Due 23rd March. Complete Ray tracer due 2nd April 2015

1 Overview

This assignment is intended to get you to understand how to write a simple ray tracer. In this process, we will also review all important techniques covered in this course. The first requirement (assignment 3 worth 10%) was to write a ray tracer that minimally ray traces a single sphere from a fixed point of view. In the second part add the ability to move the camera and add more spheres and triangle mesh objects. Add rendering features such as mirror reflections, shadows and anti-aliasing. Other enhancements are possible, and also if you add a feature not on my list then you will get credit to a maximum of 25%.

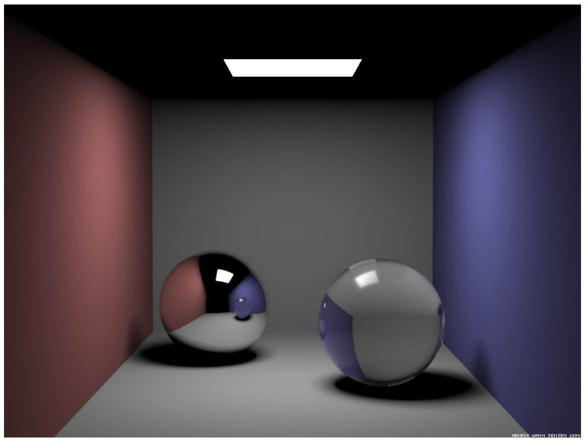

Figure 1: A image from a ray tracer. This image can be used as a high-standard reference image for your assignment.

Please also write proper documentations. At the marking demo, make an explicit list of features that you thought you have done and handle it to the TA.

2 Marking

Part one (assignment 3 sphere): 10% Part two 25% but a bonus mark will be awarded if you implement enough functionality. Features not on the list, can also be added for credit, but best to talk to the TA first.

There will also be a bottle of wine awarded to the author of the most beautiful image produced from your program.

The list of marking criteria are:

- Coding style and documentation: 2 points.
- Advanced Functionality.
  Here is a list of suggestions about what functionality to add. Feel free to invent your own features. If

you tend to stick to this list, then a good way of planning your project is pick a category from below, and try to implement some or all features in that category. The total number of points can add to 26 as there is a bonus point.

- Better Rendering
  - Shadows (3 points).
  - Reflections (4 points)
  - Refraction (5 points)
  - Multiple light sources. (3 points)
  - The ability to set an arbitrary camera view point (4 points).
- User Interface into features
  - Interactive UI for camera, material and light source parameters. (4 points)
  - Set up two view ports side by side, one renders the scene with openGL and the other with your ray tracer. Make the two view ports share the camera, light source and surface material parameters. Such that the user can move the camera around, adjust lighting and material in the openGL view port, and look at the real-time rendering feedback. Then, the user can ask the ray tracer to render with the same set of parameter, and compare the rendering result between openGL and the ray tracer. (6 points)
- Advanced Rendering
  - soft shadows (4 points)
  - Put more than one light source of a different type (for example, directional light source, or area light source). (3 points)
  - Cast sufficient number of rays so the rendered image is correctly anti-aliased. (4 points)
  - Put the sphere inside a room-like space (a cube without one side) and render the 5 visible walls with different colours. (3 points)
  - Make the sphere reflective and reflect the colour of the wall. (3 points)
- Modeling and Scene-Graph
  - Model and ray tracing a triangle mesh, for example a cube, in the same scene using a scene graph. (3 points)
  - Import a complicated triangle mesh from an external file (e.g. .obj files - such as a teapot), and ray trace it in the same scene. (3 points).
- Software Architecture
  - Write a multi-thread ray tracer which runs on several threads on the CPU at the same time. For i7 core machines, I suggest using 8 threads. (6 points)
  - Benchmark your multi-thread ray tracer. Make a chart of the rendering time of the same scene using different number of threads (from 1 thread to 8 threads, for example). Write a short paragraph which briefly explain what you have observed. (5 points)
  - implement uniform space subdivision or another acceleration scheme (6 points)