Week 1

This first exercise is primarily about getting a ray tracer up and running. If you have not previously implemented a ray tracer (including an intersection acceleration data structure) that you would like to use, we offer the following two options. (a) The course framework which has been tested over the past few years, and which has been coded to suit the exercises. (b) The framework accompanying the text book (PBRT) which already implements some of the methods to be used in the exercises, but where one must modify and use the framework rather independently to solve the exercises.

Learning Objectives

- Get a ray tracer up and running which loads Wavefront objects and renders them with Lambertian materials using directional lights or approximate area lights (objects with ambient colour are considered to be area lights).
- Be able to change the material of a Wavefront object by modifying its OBJ and MTL files.
- Render the Stanford bunny and the Cornell box with and without hard shadows.
- Explain edge aliasing and do jitter sampling to get anti-aliased edges.

Geometry and Materials

Throughout the course we expect you to be able to load triangle meshes defined by Wavefront .obj-files. This includes the .mtl-files which describe the materials attached to the different triangles of the meshes.

- Familiarise yourself with the Wavefront OBJ file format and especially with the Wavefront MTL file format. Make sure you know how to create a new material (in the .mtl-file) and how to change the material attached to an object (in the .obj-file). And make sure you understand the material parameters that you can set in the MTL format. Confer the links in the section "Reading Material" below.
- Find out how you load .obj-files into your renderer. (In the course framework, you simply provide them as command line arguments to the executables. In Visual Studio, command line arguments are provided in the project properties under Debugging.)

The models that you need for the exercises are available in the `models` folder of the course framework.

Ray Tracing

Take some time to choose which ray tracer you would like to use. Once you have made your choice, do the following.

- Load the Stanford bunny (`bunny.obj`) into your ray tracer. Preferably this should result in some simple pre-visualization of the bunny. (If you are using the course framework, use the `pathtrace` project for this exercise. Take some time to understand what you can do with the keyboard in this program. Do this by investigating the `keyboard` function in `pathtrace.cpp`.)
- Illuminate the bunny by a directional light source[1] emitting the radiance $L_e = \pi$ in the direction $\vec{\omega}_e = (-1, -1, -1)/\sqrt{3}$ (this is the default light in the framework). Make sure that you shade the bunny correctly using the Lambertian BRDF. (In the framework, you need to implement the `sample` function in `Directional.cpp` and the `shade` function in `Lambertian.cpp`.)

[1] Technically, a directional light is an infinitely distant collimated light source of infinite area. Very theoretical idea, but, from a human perspective, it somehow resembles the sun.

- Compose a new material in the file `bunny.mtl` and attach it to the bunny.
- Cast shadow rays and use extra jitter samples to get a nice anti-aliased bunny. Save the result with and without shadows (take screenshots or press 'b' on the keyboard).
- Now load the Cornell box (`CornellBox.obj`) and the blocks inside it (`CornellBlocks.obj`). The Cornell box has an area light source. Implement a simplified area light sampler which always samples the center of the area and returns the light emitted from the entire area [Larsen 2004, see reference below]. (In the framework, implement the `sample` function in `AreaLight.cpp`.[2]) Again save the resulting images.

Week 1 Deliverables

Bunny images (e.g. without shadows, with shadows, and anti-aliased), Cornell box images (e.g. without shadows, with shadows, and anti-aliased). Include relevant code and render log (number of triangles, number of samples, render time, and likewise). Please copy everything into a document and upload at CampusNet under Assignments. Consider these weekly deliverables to be your lab journal.

Reading Material

The curriculum for Week 1 is

P Sections 1–1.2. *Photorealistic Rendering and the Ray-Tracing Algorithm*.
P Chapter 5.4–5.6. *Basic Radiometry*.
P Intro. to Chapter 8 and Section 8.3. *Reflection models* and *Lambertian reflection* as a special case.
P Sections 12–12.4 except 12.2.1–12.2.3. *Light Sources*.

Alternative literature available online or uploaded to CampusNet:

- Glassner, A. S. An Overview of Ray Tracing. In *An Introduction to Ray Tracing*, Chapter 1, Morgan Kaufmann, 1989.
- Hanrahan, P. Rendering Concepts. In *Radiosity and Realistic Image Synthesis*, Chapter 2, Morgan Kaufmann, 1993.

Additional resources:

- The Wavefront OBJ file format specification: http://paulbourke.net/dataformats/obj/
- The Wavefront MTL file format specification: http://paulbourke.net/dataformats/mtl/
- The Cornell box (data and history): http://www.graphics.cornell.edu/online/box/
- Larsen, B. D. Direct Illumination. In *Real-Time Global Illumination by Simulating Photon Mapping*, Chapter 3, PhD Thesis, IMM-PHD-2004-130, Informatics and Mathematical Modelling, Technical University of Denmark, September 2004. http://www.imm.dtu.dk/pubdb/p.php?4115.

JRF 2011

[2] Hint: Look at the `AreaLight` constructor in the `realtime` project for inspiration.