Project "Computer Graphics" - Prof. Dr. Elmar Eisemann

Ray Tracing Competition

Read the assignment carefully!

The goal of this project is to create a ray tracer. As seen in class, ray tracing delivers beautiful images by recursively launching rays into the scene.

You can start with the framework downloadable via the blackboard lecture pages (or graphics.tudelft.nl/~eisemann/TI/raytracer.zip). The program provides an OpenGL interface – similar to the one you have used so far. The difference occurs, when you press the "r" key. Now, rays are shot from the current camera position into the scene and the function performRaytracing is called. In it, you should perform the raytracing against the mesh MyMesh, to determine a color, which will then be placed in the pixel corresponding to the ray. It is up to you to define what color to declare …

All the functions that need to be modified are defined in raytracing.h and implemented in raytracing.cpp. Additionally, you should look into mesh.h, to get an idea of the mesh data structure for MyMesh, a global variable that allows you to access all scene information. In raytrace.h, you find the function raytrace, which will receive one ray at a time, corresponding to a screen pixel. The color you return via this function will be placed at the corresponding pixel. Further, the file contains a function called initData(), which is called only once and that can be used to construct a potential acceleration structure for the ray tracing process, or to load and initialize textures, in case your model comes with textures (the texture name is stored in the texturename variable of the mesh's material properties). As in previous exercises, you can place (resp. add lights) using 'l' (resp. 'L'), which will also appear in a vector conveniently accessible to you.
When you export a scene via blender, the resulting .obj will come with an .mtl (material) file. The properties listed in this file will be accessible to you via the mesh structure as material elements. Feel free to give certain materials certain hard-coded properties, e.g., if material name is "mirror" you assume that the material is a mirror, independent of the other values.

Hint

Start without considering the actual scene and just intersect rays with a sphere at the origin (Exercise 3). Next, add light effects to this sphere and see if MyLightPositions behaves the way as expected. Then place a second sphere in the scene and render the first one reflected in or refracted through the second. Only then start with a simple ray tracer for triangles. Here, you can make use of the OpenGL part to 'see' what the rough shape of the result will look like, before tackling the additional effects (depth of field, shadows, motion blur…). For debugging skip every second ray by directly returning black to accelerate rendering.

Please notice that much of the code is not clean! So please do not waste your time reading through all files. Raytracing and mesh are the only files you should take a close look at.

If you notice any bugs in the code, please inform me immediately by email.

Please remember that the OBJ loading ONLY works for files directly coming from Blender. There might be OBJs online, but not all will work. You can always first import these files into Blender and then store a new OBJ version again.

Grades:

You are **not** directly judged on the quality of the rendering, but rather the amount of graphics **techniques** that you used. In other words, even if your ray tracer produces a boring image, it can still be technically impressive and lead to a good grade!
Nonetheless, there will be a **rendering competition**, meaning that the group producing the best image decided by a jury consisting of members of the Computer Graphics and Visualization Group. The best rendering receives 1.5 points extra, the second 1, the third 0.5.

Please form **groups of 7 students minimum and up to 9 total**. Recommended is 7.

Important: Any email correspondence should have as a subject "[TI1805 2014 Project] …" **When handing in your final project by email, please include a short description of your work and rendering competition entry - text maximally 150(!) words, figures as many as you like!**

Deadline is the 24th of June at 23:59 (Delft Time) – by email (assignments.eisemann@gmail.com)

Important, please report your group also via email to the above address before the 18th of June 23:59 (Delft Time).
As a subject, please use "[TI 1805 Teams]".