CS580 Homework #1 – Ray Tracing

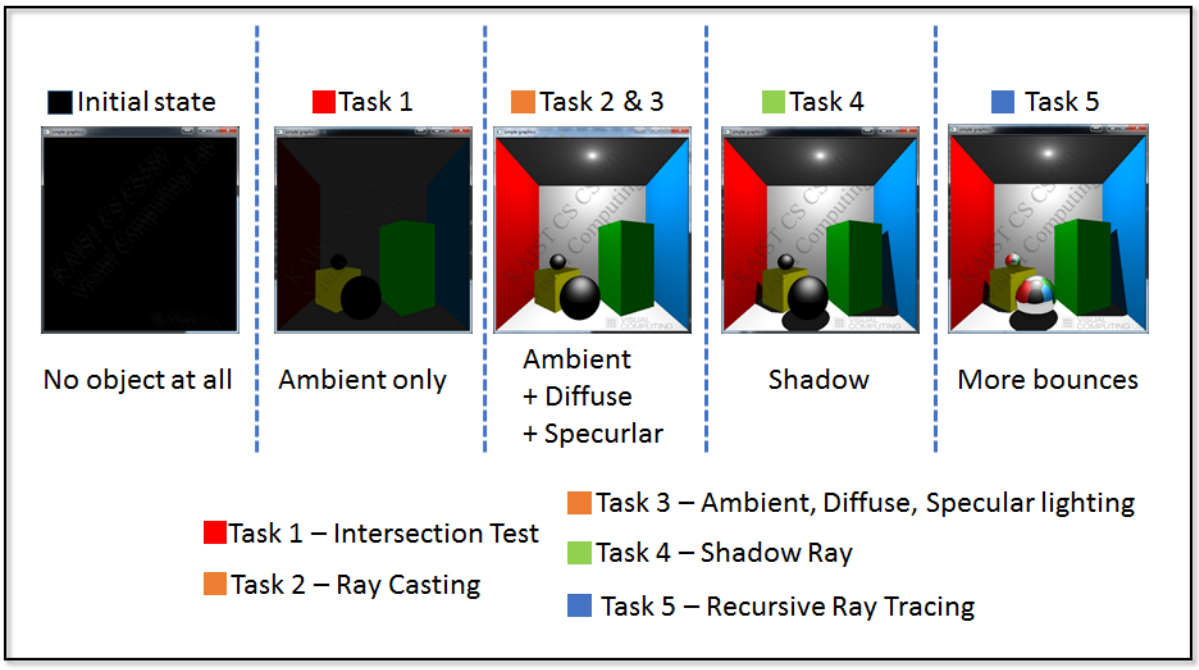

Figure 1 Overview of Tasks

1. Introduction

In the lecture we have learned Ray Tracing. It is time to make a simple ray tracer by yourself. If you fully understand ray tracing algorithm, it is not going to be hard to finish the first homework. This assignment consists of 5 tasks. Once you download, compile, and run the homework project, you will face a black window. A scene that you probably have not made with OpenGL will be shown after you finish the 5 tasks. The overview of your tasks is described in Figure 1.

2. Preparation

(1) Files

If you extract .zip file, some **.cpp** files, **.h** files, '**CS580_cornel_box.dat**', and a **VS2015 solution file** will be shown in a folder. '**.dat**' is a file that contains the scene information we will use. When opening the file with a text editor, you will see the file format description at the end of the file. You might modify it and render whatever scene you want, but your grade will be examined with the given .dat file. This .dat file should be passed as a standard input stream to the excutable. For example, in cmd.exe in Windows, "raytracing.exe < CS580_cornel_box.dat" command will pass the file as a standard input.

We recommend you work with **Visual Studio 2015**, because the codes were tested with this version. If it has not been installed on your machine yet, visit 'http://kftp.kaist.ac.kr', log in, and then you can download VS2015 with KAIST license.

(2) Freeglut

The program uses **Freeglut** library to create a window where your scene will be drawn. Please download compiled Freeglut library from

'http://freeglut.sourceforge.net/'. Then you have to configure your VS2010 so that it will know where Freeglut is placed. Follow these instructions.

(1) Setting up additional include directories

In the solution explorer, right-click the project '**raytracing**' > click '**Properties**' > Select '**C/C++**' and open up the sub-tree > Select '**General**' > You will see '**Additional Include Directories**'. Type the path of include directory of Freeglut.

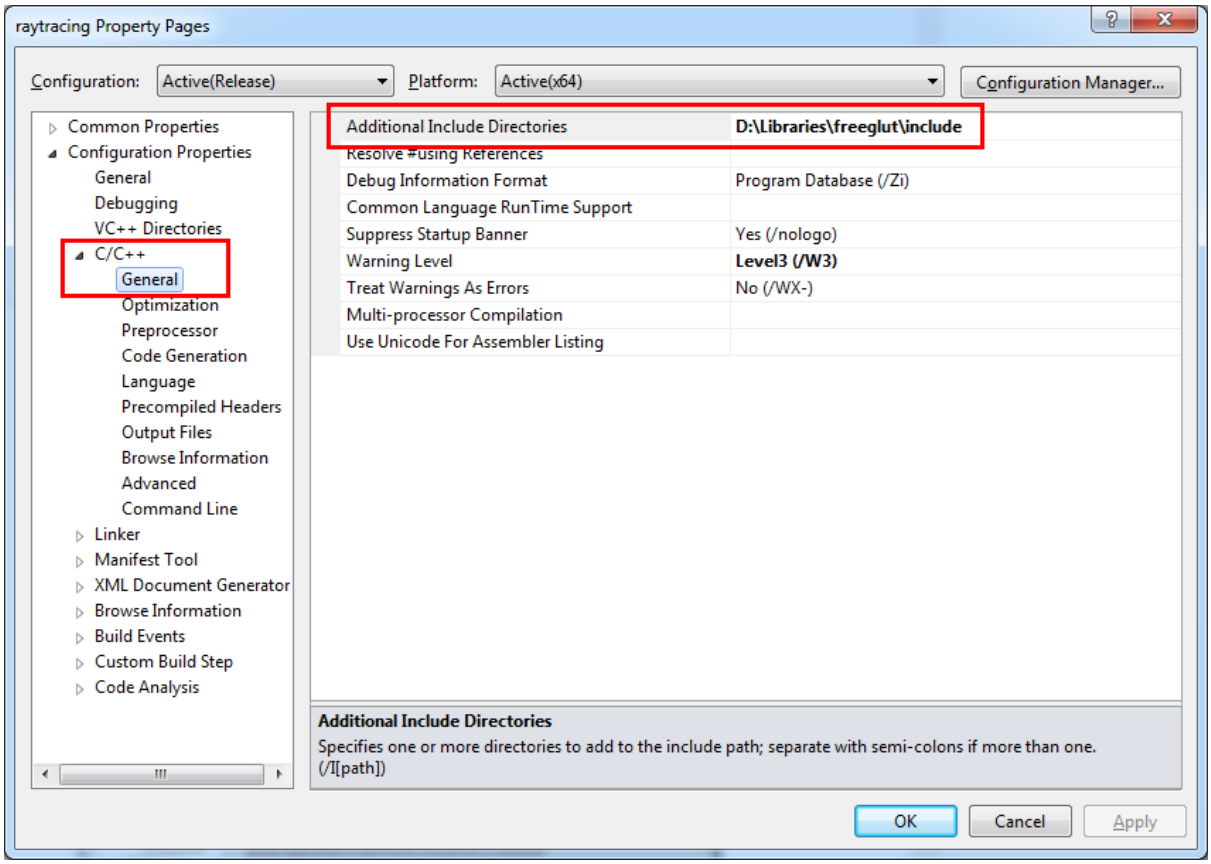

Figure 2 Setting up Additional Include Directory

(2) Setting up linker

In the solution explorer, right-click the project '**raytracing**' > click '**Properties**' > Select '**Linker**' and open up the sub-tree > Select '**General**' > Type the path of lib directory of Freeglut in '**Additional Library Directories**' > Select '**Input**' from the sub-tree > Type '**freeglut.lib**' in '**Additional Dependencies**'.

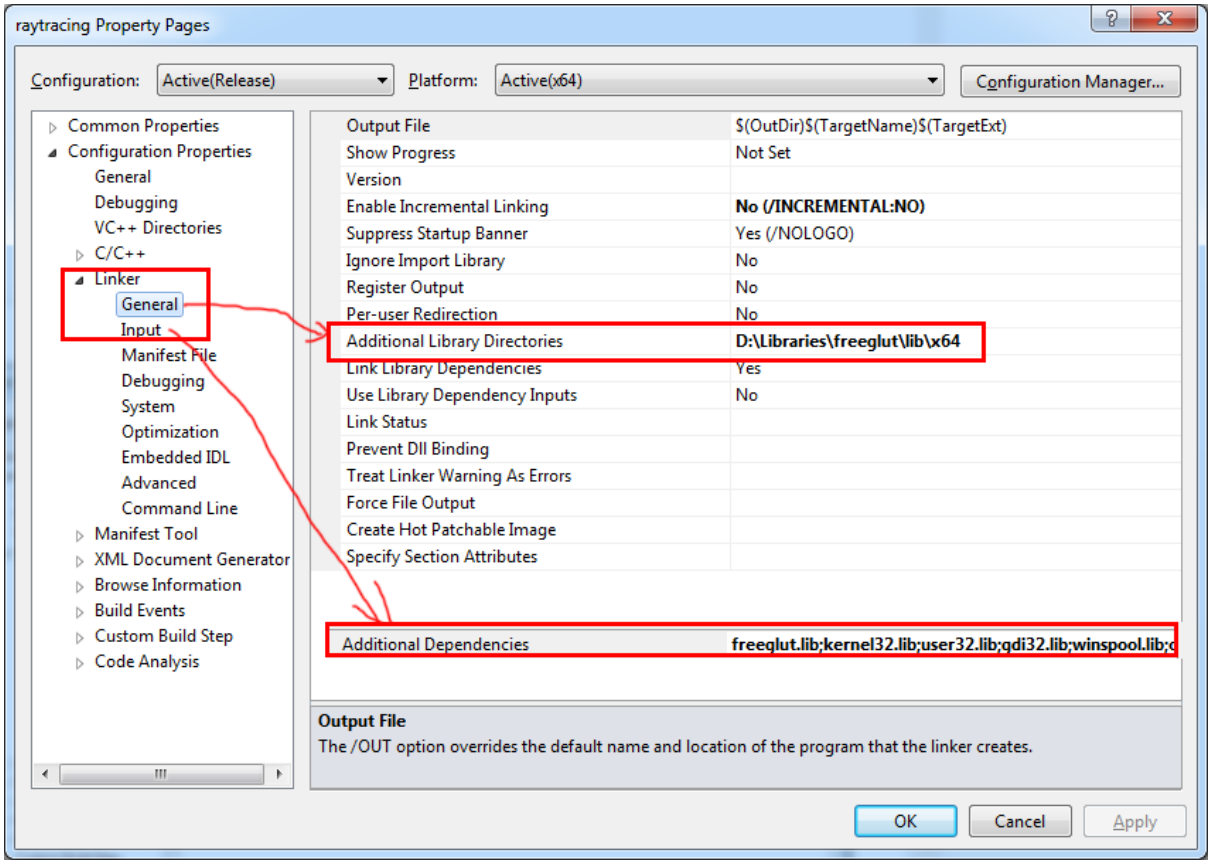

Figure 3 Setting up Linker

If you are willing to use Glut, you have to find '**#include <GL/freeglut.h>**' in the entire solution, and change it to '**#include <GL/glut.h>**'. Also, instead of 'freeglut.lib', link 'glut.lib', which you might already have or use in advance.

(3) Our program requires 'freeglut.dll' file on running time. Add the directory that contains 'freeglut.dll' file to system PATH or copy the 'freeglut.dll' file to the executable folder.

3. Tasks

You will see where to write codes if you search 'Your code' in the entire solution. Basically you are going to modify sphere.cpp, polygon.cpp, and litscene.cpp, but you have to read all the source code to understand the structure of the project.

(1) Task 1 – Intersection Test

If you manage to compile the project, you will see a black image on a window. This is because two intersect() functions in polygon.cpp and sphere.cpp are always returning false for now. Therefore, the ray tracer thinks that there is no object in a scene. Your job is to implement the intersect() functions in polygon.cpp and sphere.cpp. Once you complete these functions you will see the scene lit under only ambient lights. In the

overview, only two boxes are shown, because the ambient reflectance of other objects are set to (0, 0, 0).

(2) Task 2 – Ray Casting & Task 3 – Ambient, Diffuse, Specular lighting

Complete LitScene::colourOnObject() in litscene.cpp. For now only ambient color is computed. You must add diffuse and specular color to the variable named colour. We recommend you to use Blinn–Phong shading model for specular reflection. After Task 2&3 you will face a more colorful scene.

(3) Task 4 – Shadow Ray

Modify LitScene::colourOnObject() in litscene.cpp to bring shadow effects into your scene. As you learned in the course, you have to generate a shadow ray, which penetrates from a point to a light source. If the shadow ray intersect with an object (sphere or polygon), the point is occluded and must be under shadow.

(4) Task 5 – Recursive Ray Tracing

In LitScene::intersect(), the color of the first bounce ray is taken. Fix this part in order to consider multiple bounces. After that you will see the metal spheres reflecting the environment in the scene.

4. Scores

Tasks	Scores
Task 1 – Intersection Test	10 pts
Task 2 – Ray Casting	10 pts
Task 3 – Ambient, Diffuse, Specular lighting	10 pts
Task 4 – Shadow Ray	10 pts
Task 5 – Recursive Ray Tracing	10 pts

5. Submission

The deadline is 23:55 (5 minutes before the midnight), 7th April 2017. Pack your source codes and documentation about the homework in .zip. The PDF formatted documentation must include some screenshots of 5 tasks. You must **NOT** include executable files and library files in .zip file. Send your homework by email to gjnam@vclab.kaist.ac.kr. *Your questions are welcome.*