Basic 3D Computer Graphics – Assignment

Due Date: as announced in lectures

In this assignment you are asked to write and test a basic ray tracer for spheres and planar disks. have A ray tracing skeleton is provided to start with. It defines the required user interface and reads scene information from a file (format described overleaf). It also sets up many useful classes. This is available from the file `AssWorkspace.zip`. The skeleton will compile and run and produce a splash of colours in a GL window.

Testing is part of this assignment. You need to design your own test scenes and decide whether your results are correct. Once your ray tracer is working, design an original demonstration scene of at least moderate complexity, in a file called `myScene.txt`.

You are also to include in your submission a report that describes your demonstration scene and the most interesting bug that you found while developing your program. For the bug, describe how you discovered it and how you fixed it. If your ray tracer has any known deficiencies at the time of submission, list these in the report too. Reports should be in a plain text file called `myReport.txt` and be no more than 66 80-column lines long.

Your ray tracer, report, and scene file are to be your own work. The only exceptions are that you may include source code from our lecture notes or from the required textbook (Hill, Computer Graphics using Open GL, 2nd ed., Prentice Hall, 2000, or 3rd ed., 2007). It is not recommended to copy source code from the textbook, however. You would not learn much from copying and the ray tracer described in the book is much more complex than is necessary for the assignment.

Marking Scheme

This assignment is worth 30% of your marks for the course. It will be marked out of 100 as follows:

25 Marks	Basic ray intersection showing colors and correct geometry of spheres with ambient light for an arbitrary view point on the negative *z* axis
15 Marks	Lambertian and Phong illumination
15 Marks	Shadows cast from point light sources
15 Marks	Mirror reflections
15 Marks	Program also works for disks
15 Marks	Report and scene file

High marks will be given for a basic ray tracer that implements the above features and which is easy to follow. You will not get any extra credit for extravagant solutions. In fact, *you will lose marks if you add features to your program that changes the expected input or output of the ray tracer*.

Assignment submission

All files should include your name and ID in a comment at the beginning of the file. All your files should be able to be compiled without any editing. All files should include adequate documentation.

The assignment due date is as announced in lectures.

Please Submit :

- All source files relating to your ray tracer
- An original test scene file called `myScene.txt`
- A report file called `myReport.txt`

Scene Description file Format

The scene description file is a plain ASCII text file. Each line of the scene description file will be one of the following:

# text	The character '#' followed by arbitrary text. These lines are for comments and are ignored.
view *s d r k r g b*	The text 'view' followed by parameters relating to the scene. If there is more than one 'view' line, only the last one is used. The eye point is at (0, 0, -*d*) and the screen extends from (-*s*, -*s*, 0) to (*s*, *s*, 0). The "up" vector is (1, 0, 0) and a right handed coordinate system is used. The output window is *r* pixels wide and *r* pixels high. One ray should be fired through the center of each pixel. The integer *k* determines the maximum number of mirror reflections that a single ray can go though. The tuple (*r*, *g*, *b*) gives the red, green and blue intensity of the background. This is the colour returned by a ray that does not hit any of the objects in the scene.
ambient *r g b*	The tuple (*r*, *g*, *b*) gives the red, green and blue intensities of the ambient light. If more than one *ambient* line is found, only the last one is used.
light *x y z r g b*	The word 'light' followed by real numbers. The point (*x*, *y*, *z*) gives the location of a point light source and the tuple (*r*, *g*, *b*) gives its red, green, and blue intensity in the range 0..1. The intensity of the light source does not diminish with distance. In principle, there can be any number of lights, but we will test your program with five or fewer of them.
sphere *x y z r dr dg db sr sg sb f*	The word 'sphere' followed by real numbers. A hollow sphere of radius r is at the point (*x*, *y*, *z*). Its diffuse and ambient reflectivities are defined by red, green and blue components (*dr*, *dg*, *db*) in the range 0..1. Its specular colour is defined by (*sr*, *sg*, *sb*). These values are to be used for mirror reflection and Phong illumination. The Phong illumination exponent parameter is defined by *f*. In principle, there can be any number of spheres, but we will test your program with ten or fewer of them.
disk x *y z nx ny nz r dr dg db sr sg sb f*	The word 'disk' followed by real numbers. A disk is a circular region of a plane. The disk is centred at (*x*, *y*, *z*) with radius *r* and surface normal (*n1, n2, n3*). The parameters *dr*, *dg*, *db*, *sr*, *sg*, *sb, and f* are as for spheres. In principle, there can be any number of disks, but we will test your program with ten or fewer of them.
end	The word 'end' followed by arbitrary text. All lines after this one are ignored. All scene description files must have an 'end' line.