# Ray Tracing My Workspace

Yu Yi Lin

yylin

20330220

**March 4, 2012**

## Purpose:

The purpose of this project is to provide extra functionalities to the previous ray-tracing scene to display a more realistic scene to the real world.

## Topics:

- Provide additional primitives and various operations to construct more complex geometry and also provide texture mapping for realistic model.
- Adding real world effects to provide realistic scene especially for transparent objects.

## Statement

The scene that I am going to ray trace is my workspace. The first thing to mention is the transparent glass I have for drinks on the side of my desk. Next, it would be the 21-inch desktop monitor I have for doing all the high intensity and tears-dropping CS assignments (and you know what course I am referring to). Again the edge of the monitor has a glassy surface which will contain all sort of reflection of other objects on the desk together with the keyboard and mice. The last thing to mention would be the lamp of the top left of the workspace to provide lighting.

Now in order to construct the above scene, I would have to design each objects individually and later combine everything together. The scene will be challenging because of multiple objects especially the transparent objects. Things that I will get from the project are the various methods that attempt to make the 3D model in the computer present more closely to the real world and how to do these methods in an efficient way.

## Technical Outline

Here I will go over the technical details of this project in top-down order of the objective list. After going over some search, I've found using the Cornell Box is a good setting for testing rendering scene so I would set it up as the basic environment to test each objective before putting everything together. I've implemented anti-aliasing using simple anti-aliasing in assignment 4 so I would try to improve it by implementing anti-aliasing using stochastic sampling.

The additional primitives I will be adding are cone and cylinder. Thus the interaction function on these two primitives will be added and remember to add new commands to the lua script as *gr_cylinder* and *gr_cone*. The next objective – texture mapping would be done by loading up a png file and use the RGB of the picture as primitive ray to trace and the picture can be set by using the new commands *setSurface*. After finishing the basic primitives, constructive

solid geometry should be done by using ray casting together with hierarchical tree structure, three new lua commands would have to be added – *union, intersection* and *difference*. These would be considered as basic setup of the project so they have to be done first.

After getting all the work on primitives done, photon mapping should now be added in order to make further illumination effects on the objects (refraction, reflection and transparency). **This will be expected to be the hardest part of the project**. There are two-passes to accomplish during photon mapping – photon tracing and rendering. During the photon tracing pass, the first thing is to determine photon emission of light source. Next is photon scattering which determines how many photons are absorbed, reflected or refracted by each object (which will be objective 6,7 and 8), this should be done using Russian Roulette where we probabilistically decide whether or not to absorb, reflect and refract a proton. One big problem now is how to store these protons and naive solution would be in any basic data structure like array or vector. However to reduce the space and improve the performance of the neighbour searching process, kd-tree is commonly deployed. The second pass of the photon mapping is rendering. The rendering process is all about approximating the rendering equation which is also a fairly difficult part of the mapping.

I've also discovered that the speed of ray tracing is extremely slow thus I would be dividing the ray tracing process into multiple threads that would improve the performance by dividing the image into equal size region for each thread to trace.

The last bit of work to make the scene better is to implement anti-aliasing using adaptive supersampling. It splits the pixel into four portions and ray cast through the corners of the four quadrants and instead of averaging all the samples, the method only divide the quadrants with significant difference and ray cast these quadrants, this then continue to some level.

## Bibliography

Cooksey, Chris. *Antialiasing and Raytracing*. N.p., Jan. 1994. Web. 3 Mar. 2012. <http://paulbourke.net/miscellaneous/aliasing/>.

*Jensen, Henrik W.*, Realistic Image Synthesis Using Photon Mapping, A K Peters, Ltd., Massachusetts, 2001

*James T. Kajiya.* The Rendering Equation**.** Computer Graphics (Proceedings of SIGGRAPH 86), August 1986, pp. 143-150.

*Geoff Wyvill, Tosiyasu L. Kunii.* A Functional Model for Constructive Solid Geometry. The Visual Computer*, 1(1),* July 1985, pp. 3-14.

Hearn, Donald, M.Pauline Baker, and Warren R. Carithers. *Computer Graphics with OpenGL*. 4th ed. N.p.: Prentice hall, 2011. 183-85, 401-07,523, 565-75, 661-62, 673-93. Print.

Molofee, Jeff. *NeHe Productions: Texture Mapping*. GaveDev.net, n.d. Web. 3 Mar. 2012. <http://nehe.gamedev.net/tutorial/texture_mapping/12038/>.

*The Cornell Box*. Cornell University Program of Computer Graphics, 2 Jan. 1998. Web. 3 Mar. 2012. <http://www.graphics.cornell.edu/online/box/>.

Wolfe, Rosalee. *Teaching Texture Mapping Visually*. Ed. Scott G. Owen. DePaul University, 30 May 1999. Web. 3 Mar. 2012. <http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_1.htm>.

## Documentation

A ***README*** file will be under the ***A5*** directory following the template.

## Sources

All source files will be saved under the ***A5/src*** directory and the ***make*** command will compile the files.

## Executable

The executable will be named ***scene***.

## Data Files

All data files will be saved under ***A5/data*** directory.

## Objectives:

**Name:** Yu Yi Lin

**UserID:** yylin

**Student ID:** 20330220

__ Objectives 1: Additional Primitives: Cone and Cylinders are added.

__ Objectives 2: Texture mapping with pattern or pictures work on to the basic and additional primitives.

__ Objectives 3: Constructive Solid Geometry can be done on all primitives.

__ Objectives 4: Photon Mapping is added.

__ Objectives 5: Kd-tree for photon mapping to improve performance.

__ Objectives 6: Refraction is added.

__ Objectives 7: Reflection is added.

__ Objectives 8: Improve performance by multi-threading.

__ Objectives 9: Anti-aliasing by stochastic sampling.

__ Objectives 10: Final Scene

**Declaration:**

I have read the statements regarding cheating in the CS488/688 course handouts. I affirm with my signature that I have worked out my own solution to this assignment, and the code I am handing in is my own.

**Signature:**