Spring 2006

22C:196 Advanced OpenGL Rendering

Assignment 4

Due: Tuesday, April 18th at 11:59pm

Goal: Implement basic reflection (and perhaps refraction) using OpenGL.

Problem 1 (15 points): Write a simple OpenGL program that reflects a scene around a simple, planar mirror. The scene should be an enclosed room, with one mirrored wall and the remaining surfaces textured. Include one complex object (e.g., Al, a teapot, F-16, etc) that can be moved around the scene via the mouse. Note:

- Make sure you add a clipping plane at the mirror's surface, and make sure I can easily move your complex object to test this functionality.
- The room should have (at least) one light that affects reflected geometry correctly. The user should be able to move the light around the room.
- You may wish to subdivde the textured walls in your scene into multiple triangles (about 400-900 each) so the lighting on the walls looks more realistic.
- You may use either a stencil-buffer or render-to-texture approach to this problem.

Problem 2 (35 points): The idea of this problem is to extend your mirrored-room scene into an interesting demo you can show off to people and put on your webpage. Your result should be something like the mirrored-room demo I showed the first day of class, but you may choose which of the following features to implement as long as they total (at least) 35 points:

(15 points): Add an additional mirror, and allow recursive reflections between the mirrors. Add a menu or key command that allows the user to increase or decrease the maximum recursion level. Make sure the program starts with a view that can see recursive reflections (perhaps by having a non-rectangular room).

(5 points): Make (at least) one of your mirrors have an interesting shape (e.g., a teapot, bunny, or dragon shaped mirror).

(5 points): Add a user interface to allow movement (of the view) around the room via keyboard or mouse commands.

(10 points): Make your floor reflective, but only partially reflective (so you can see the floor texture in addition to the reflection). The reflectivity should vary based upon the angle between the unit-length surface normal and the normalized viewing vector: reflectivity = $(1 - \vec{N} \cdot \vec{V})^5$. Use a Cg program to control this blending. *This is a Fresnel effect, and the equation is Schlick's approximation. Note the value should be in the range [0..1], which means* $\vec{N} \cdot \vec{V} \in [0..1]$.

(20 points): Implement refraction for the complex 3D model (e.g., the bunny, dragon, or triceratops models). In order to see the room refracted through the object, you should intersect the refracted ray with each wall in the room (in your fragment program). If you have other non-planar objects in the room, you may remove them for this part of the assignment. Any mirrors need not reflect the refractive object. Add a menu or key command to toggle the object between refractive and opaque. Use an index of refraction between 1.2 and 1.4.

(10 points): If you add a refractive object, allow the mirror(s) to reflect it.

(15 points): Add shadows to your scene. You may use any shadowing technique you want, but shadows should appear on (at least) all walls, the floor, and the ceiling, and must be reflected (and refracted) by any mirrors (or refractive objects).

(5 points): Add a toggle that applies an image processing effect (from the previous assignment) to the mirrored reflection. For instance, the reflection could have a gaussian blur or reflect a grayscale image.

(Optional) Problem 3 (50 points): Instead of doing Problems 1 and 2, you may implement explosion mapping entirely on the graphics card. If you choose to do this, please come talk to me first.