Fall 2007

22C:151 Introduction to Computer Graphics

Assignment 10

Due: Tuesday December 5th at 11:59 pm

Goal: Extend your basic raytracer to handle shadows, reflections, refractions, and triangles.

Problem 1 (5 points): Extend your ray tracer to compute shadows on Lambertian surfaces. Inside your *LambertianMaterial::Shade()* routine, check if the light is visible from the hitpoint (by shooting another ray towards the light). If the light is visible, shade as normal. If the point is shadowed, then either return black or an ambient color.

Problem 2 (5 points): Add a reflective material that behaves like a perfect mirror. *(In other words, write a "ReflectiveMaterial" class that inherits from the "Material" class.)*

Problem 3 (10 points): Add a refractive material that transmits light. *(In other words, write a "RefractiveMaterial" class that inherits from the "Material" class.)*

Problem 4 (5 points): Add a triangle object type to your ray tracer. *(In other words, write a "Triangle" class that inherits from the "Primitive" class.)*

Problem 5 (5 points): Read an OBJ file to your raytracer and render it using a diffuse material type. Remember OBJ files consist of a number of triangles, the primitive you added to your ray tracer in Problem 4.

> *Hint: Look at the 'glm.c' file in order to figure out how the triangle data is loaded into memory.*
>
> *Hint: If you use large models, your ray tracer will run very, very, very slowly. Use a smaller model, like Al Capone.*
>
> *Hint: If you plan on doing this part late on the evening of Wednesday the 5th, your program will not finish executing in order to submit your homework on time! You are responsible for such delays, so start early!*

Extra Credit (5 points): Modify your ray tracer into a path tracer. Your need only worry about diffuse surfaces and you can assume the only light source is a (non-black) constant color background. You may wish to create a *PathtracedLambertianMaterial* class to differentiate it from the ray traced version.

NOTE: A "README" file is required in order to get full credit! Please tell us how to compile and run your program. Additionally, tell us how we can recreate images that demonstrate all five parts of the homework! If we cannot recreate an image exhibiting the results of a problem, you will not get credit for it!

NOTE: Example images and scene data to recreate them will be posted online.