Group Members: Joshua Sonnen, Jackson Sweet

Project Manager: Joshua Sonnen

Final Deliverable:

All code presented here builds from our A2 raytracer.

Area lights and soft shadows– Area lighting can be implemented by creating a rectangular area light object. Each object will take parameters for size, texture, texture page vertices, and brightness. Lighting calculations will be performed by sampling from several rectangular subsections of the area light to calculate the fraction of the light visible to a given pixel. A global shading parameter for the number of samples may be implemented to increase quality at the expense of performance; the number of samples should always be some integer square (1, 4, 9, etc.). Another global parameter may be implemented to switch between regular, random, and stratified sampling for the light sources.

Aliasing– average rays across sub pixels to "fake" a higher quality resolution.

Edge Detect– I only want to alias at the edges (as viewed by the camera). Because I think that blocky edges look bad. As I iterate through my rays, I will check object ids. If the object id of closest hitrec changes, then I know I have hit an edge. There will be a locality parameter (distance in pixels around an edge to alias).

Object Alias– Maybe the user wants aliasing only on certain objects. One object is a particularly small triangle mesh or far from the camera.

Add light interactions for triangle meshes– I recently learned about flat, Gouround, and Phong shading in class. I will implement all 3 of these to be compatible with any triangle mesh loaded from an OBJ file format.

Refraction– each object will have a refraction and reflection coefficient. Use Snell's law to find the angle of refraction. Then shoot 2 rays: 1 reflected and 1 refracted. Use refraction coeff and reflection coeff to weight each ray component before summing them.

Expansions–

- Are there any other ways in which light interacts with a surface to produce an effect not implemented in A2?
- Supersampling for area lighting and soft shadows may be expanded to incorporate antialiasing

Milestone 1: Jackson will write the area lights object. It will be represented as a rectangle of point light sources. These point lights can be sampled uniformly, randomly, or using a stratified approach. Josh's implementation of all 3 lighting options for triangle meshes should be complete, as well (flat, Gouround, and Phong).

Milestone 2: Add doc strings. Rigorously test all functions completed from Milestone 1. Add additional functionality to our area light object. Josh will add a refraction coefficient to our scene objects. Find some awesome triangle mesh online to create an interesting scene.

Roadmap:

Jackson– Create a pseudocode template and mathematical framework for area lights and soft shadows. Begin writing code to complete this subtask by Milestone 2.

- Area light will be another lighting object
    - Parameters– length, width, intensity
    - Subpixel sampling – uniform, random, and stratified

Josh– 3 lighting components

- Flat
- Gouround
- Interpolated (Phong)

Josh– Refractions. Each object in the scene will have reflection and refraction parameters. At a ray intersection, calculate the reflected ray and the refracted ray. Shoot those rays into the scene using traceray. Remember to set a maximum recursion depth.

Josh– Edge Detection

Resources:

None