CSE168: Rendering Algorithms
Programming Assignment 3

Due 11:59pm, Tuesday, May 18

Henrik Wann Jensen
Toshiya Hachisuka

This assignment involves implementing shading models as well as path tracing. Percentages indicate percentage value for this assignment.

Task 1: Implement a shading model: 25 %
Implement a shading model that handles:

- Diffuse shading
- Specular reflection
- Specular refraction
- Specular highlights (Phong model)

Task 2: Implement a stone texture: 25%
Implement a texture that uses cellular texturing to make a stone pattern. Within each stone pattern use a regular noise texture to add variation. You can use the provided PerlinNoise and WorleyNoise classes. Render an example image.

Task 3: Implement square area lights: 20%
Implement a square area light source that uses n samples to compute soft shadows. Render an example with soft shadows.

Task 4: Implement path tracing: 20%
Implement path tracing. Use multiple samples per pixel. Simulate indirect diffuse lighting by tracing diffusely reflected rays (you may extend the shader written in Task 1). Render the box scene.

Task 5: Implement HDR environment mapping: 10%
Add support for high dynamic range (HDR) environment mapping. You can load .pfm HDR images using the provided code. Render an image with high dynamic range lighting. Render an object with either specular reflection or specular reflection using an environment map.

Hacker points: Implement image based lighting: +10 %
Use an HDR environment map as a set of directional light sources. Render a diffuse object where an HDR environment map is the only light sources.

Hacker points: Implement bump mapping: +10 %
Add support for bump mapped procedural textures. Render the stone texture from task 2 with bump mapping.

Turnin

Your archive should contain images that demonstrate your results of the tasks, the code, and intersection statistics.
Resolution should be 512^2. Do not include any obj files in your submission. You should package up all source files and project files into a `zip` archive. Do not include any of the provided `.obj` files or any generated files from compilation (i.e. clean your project). You should only need to include the `.cpp`/`.h` files, the `.sln` and `.vcproj` Visual Studio files, plus the Makefiles if any. If your code archive is larger than 1MB, you might be doing something wrong. Note that the grading itself is done on-site, so just sending the code is not enough for get grading.
Your archive should be named according to your first initial and last name. For instance, John Smith should submit `jsmith.zip`. Convert all your images to PNG format before submission and include them in your archive. Prefix your images using the same naming convention.
Email your archive to cse168-turnin@graphics.ucsd.edu before the deadline and before you get graded.