Worksheet 7

We will add some final touches to make the renderer fully capable of rendering path traced images and using 360 degrees photographs for environment lighting. This addition of the ability to render soft shadows makes the renderer more production relevant.

Learning Objectives

- Monte Carlo evaluation of direct illumination
- Sample points on a triangle mesh.
- Using a panoramic texture to insert a background environment map.
- Render soft shadows including the effects of environment lighting.

Ray Tracing

In Worksheet 4, the Lambertian shading for area lights was only an approximation. Monte Carlo techniques can eliminate this inaccuracy.

- Load the Cornell box (`CornellBox.obj`) and the blocks inside it (`CornellBlocks.obj`) into your ray tracer. Instead of the simplified area light sampler used in Worksheet 4, implement true sampling of a random position on the surface of the area light (first sample a random triangle, then sample a random position on that triangle). Update the `sample` function in `AreaLight.cpp` and, if necessary, the `shade` function in `Lambertian.cpp`. As a result you should obtain an image of the Cornell box where the blocks cast soft shadows. Try loading a mirror ball (`CornellLeftSphere.obj`) and a glass ball (`CornellRightSphere.obj`) instead of the blocks and see if you get caustic illumination.
- Load the Stanford bunny or the Newell teapot. Go to HDRI haven (https://hdrihaven.com/) and choose a light probe that you would like to use as your environment map. Load it into your ray tracer by setting the `bgtex_filename` variable in the `Initialization` section of `RenderEngine.cpp`. If the loading fails, check that the path to the image file is correct. If loading still fails, try opening the image file in a simple image viewer (like Paint) and save the image without any modification. Ensure that your ray tracer makes a look-up into the environment map when no geometry is encountered. Change the render resolution in `RenderEngine.cpp` to widen the image and switch off the default light source (set `use_default_light` to `false`).
- Once the background environment texture is loaded, select a good scene configuration and store the camera settings. In the framework, zoom by pressing 'z' or 'Z' on the keyboard to adjust how much of the background you see behind the object, and use the virtual trackball attached to the mouse to position the object in the environment. Save your camera settings by pressing 'S'. Load the camera settings after restarting the program by pressing 'L'.
- Insert a holdout plane in the `init_tracer` function of `RenderEngine.cpp` and implement a holdout shader to let the loaded object cast shadows onto the ground in the environment map. Do this by implementing the `shade` function of the file `Holdout.cpp`. The holdout shader should compute ambient occlusion by tracing rays in directions sampled on the hemisphere over each surface point. The idea is essentially that the environment (the background) is an infinitely distant ambient area light.

Worksheet 7 Deliverables

Cornell box image with blocks that cast soft shadows, and an image with mirror and glass balls instead of the blocks. An image of a bunny or a teapot placed in a photographed environment where it casts soft shadows onto the ground. Please insert code snippets and images into your lab journal.

Reading Material

The curriculum for Worksheet 7 is

- **B** Chapter 13. *Sampling*.
- **B** Section 11.4.5. *Environment Maps*.
- **B** Section 19.2.3, 20.2–20.3. *Dynamic Range*.

Additional resources:

- Landis, H. Production-ready global illumination. In RenderMan in Production, ACM SIGGRAPH 2002 Course Notes, Chapter 5, pp. 87-101, 2002.
- Pharr, M., and Green, S. Ambient Occlusion. In *GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics*, Chapter 17, Addison-Wesley, 2004.
  https://developer.download.nvidia.com/books/HTML/gpugems/gpugems_ch17.html