Photon
CS224 Project 4
Out: Thursday, February 19
Due: Thursday, March 5

Fig. 1: Ratatouille kitchen rendered using photon maps (Copyright 2007 Disney/Pixar). Interested in reading about how Pixar uses photon mapping? See chapter 5 of “Photon Mapping for Complex Scenes:” [http://graphics.pixar.com/library/HQRenderingCourse/paper.pdf](http://graphics.pixar.com/library/HQRenderingCourse/paper.pdf)

1 Introduction

In this assignment, you will add photon mapping support to a Monte Carlo raytracer. Photon mapping is a global illumination technique that separates scene geometry from radiance calculation. The algorithm is completed in two passes. First, radiance from the light, split into discrete chunks called “photons,” is transported through the scene, and is saved in a 3D spatial acceleration data structure called a photon map. Second, the renderer queries the photon map for nearby photons, and uses them to compute an estimate for indirect illumination.

Photon mapping can be integrated with just about any rendering algorithm. When properly tuned, photon maps produce results that look visually similar to techniques like pathtracing, without requiring as much of a wait. However, they do not converge to the correct solution, and will look wrong if not tuned properly.

2 Requirements

Your final image must support full global illumination, with caustics from reflection/refraction and color bleeding from diffuse interreflection. Specifically for a C, you must have

- Recursive Monte Carlo raytracing, with support for reflection, refraction and soft shadows (already provided in support code)
- Separate direct illumination calculation
- Photon map generation, by tracing photons through the scene using Monte Carlo techniques
- Final radiance calculation combining Monte Carlo raytracing and photon map estimate

For a B grade, implement final gathering which will drastically improve image quality at the cost of speed. See section 32.6.2 in the book for details.

For an A grade, you may implement one of the following:

- Image-Space Photon Mapping
- Participating Media (using a photon map)
- Shadow Photons
- VP Trees for storing the photon map
- Importance Sampling in final gather step
- Subsurface Scattering
- Irradiance Caching
- Something else! (But check with a TA first)

To run the demo implementation of photon, type `photon_demo` in a terminal. Your results, minus any extra effects, should match the demo when the same rendering parameters and scene file are used.

3 Getting Started


2. Support code is available in `/course/cs224/asgn/photon`

As previously mentioned, the support code already implements a Monte Carlo raytracer. You will implement the `scatter()` method, which shoots a photon into the scene, and `diffuse()`, which uses the photon map to estimate the radiance due to diffuse inter-object reflection (hence the name). Short hints have been left as comments in the support code. We also suggest reading and understanding `trace()`, the recursive raytracing function, as well as the subroutines it calls directly.

Rendering parameters used by the support code are provided as constants in `app.h`. The default parameters were chosen to balance between rendering speed and picture quality. You can get better-looking pictures by increasing the photon count and/or increasing the number of direct lighting samples taken per pixel. However, if you do this you’ll be waiting a long time to get your picture!

By default, the support code shows you your photon map as it is built, and then shows the final rendering as the raytracer runs. To view just the photon map, press the 1 key on your keyboard; to view just the rendering, press 2.

As in the path tracing assignment, we recommend referring to the “Rendering in Practice” chapter of the course text (Ch. 32). The example code given in the book uses G3D 8.x, whereas the support code uses G3D 9.x, but the concepts carry over.

You may find the G3D9 Surfel::scatter() method useful in this assignment. In Path, you called it with the enum G3D::EYE_TO_SOURCE, you’ll want to use the SOURCE_TO_EYE when doing scattering for photon mapping.

Note: students last year found this assignment fairly easy, compared to Path or As-Rigid-As-Possible. That said, please start this early! The next project will be released before this assignment is due. This is because we’re giving you a lot more time to work on projects - by overlapping assignments
a bit, you'll have access to as many TA hours as possible. You can use this opportunity to try something really ambitious or creative for extra credit. In fact, if you want to try something super cool but risky, we would be more than happy to extend your deadline to work on Photon (but we ask that you submit the B-grade stuff before the deadline).

4 Handing In

Hand in your assignment using the command `cs224_handin photon`. Remember to document design decisions, known bugs and other useful tidbits in your README. Do not hard-code any paths in your program!