Global Illumination Ray Tracer

Ioannis Tsiompikas

nuclear@member.fsf.org

May 27, 2009

Global Illumination

Definition

Simulation of the distribution of light in a 3D environment, including such effects as:

- Reflection & refraction
- Diffuse inter-reflections
- Caustics

Rendering equation

$$I(x, x') = g(x, x') \left[e(x, x') + \int_S \rho(x, x', x'') I(x', x'') dx'' \right]$$

Algorithms

Finite Elements

- Radiosity

Monte Carlo

- Path tracing
- Bidirectional path tracing
- Photon mapping
- Metropolis light transport

Photon Mapping

Two-pass global illumination algorithm.

Pass 1: photon tracing

- Shoot photons from light sources
- Bounce photons off or store at intersections

Photons are stored in a separate spatial data structure, decoupled from geometry (generally balanced kd-trees).

Pass 2: rendering

Regular ray tracing, at each intersection the photon map is used to calculate radiance using either:

- Radiance estimate directly from the photon map
- Monte carlo ray tracing

Project Overview

Description

Implementation of a full global illumination ray tracer, using photon mapping.

The major goal of the project is to be able to simulate all possible light paths through a 3D environment (`L(S|D)*E`). In essence the following should be supported:

- Indirect illumination
- Color bleeding
- Caustics
- Reflection & refraction

Additionally the following will be implemented:

- Soft shadows (area lights)
- Glossy (imperfect) reflection & refraction
- Motion blur

Design & Implementation

- Implementation in C++
- Portable across UNIX systems (GNU/Linux, BSD, IRIX, MacOS X), and windows through cygwin.
- Multithreaded.
- XML scene description (with converters for various common scene formats).
- Output of single images and animations.

Testing Strategy

Correctness

- Visual inspection
- Comparison with path tracing
- Comparison with other renderers (e.g. RADIANCE).

Efficiency

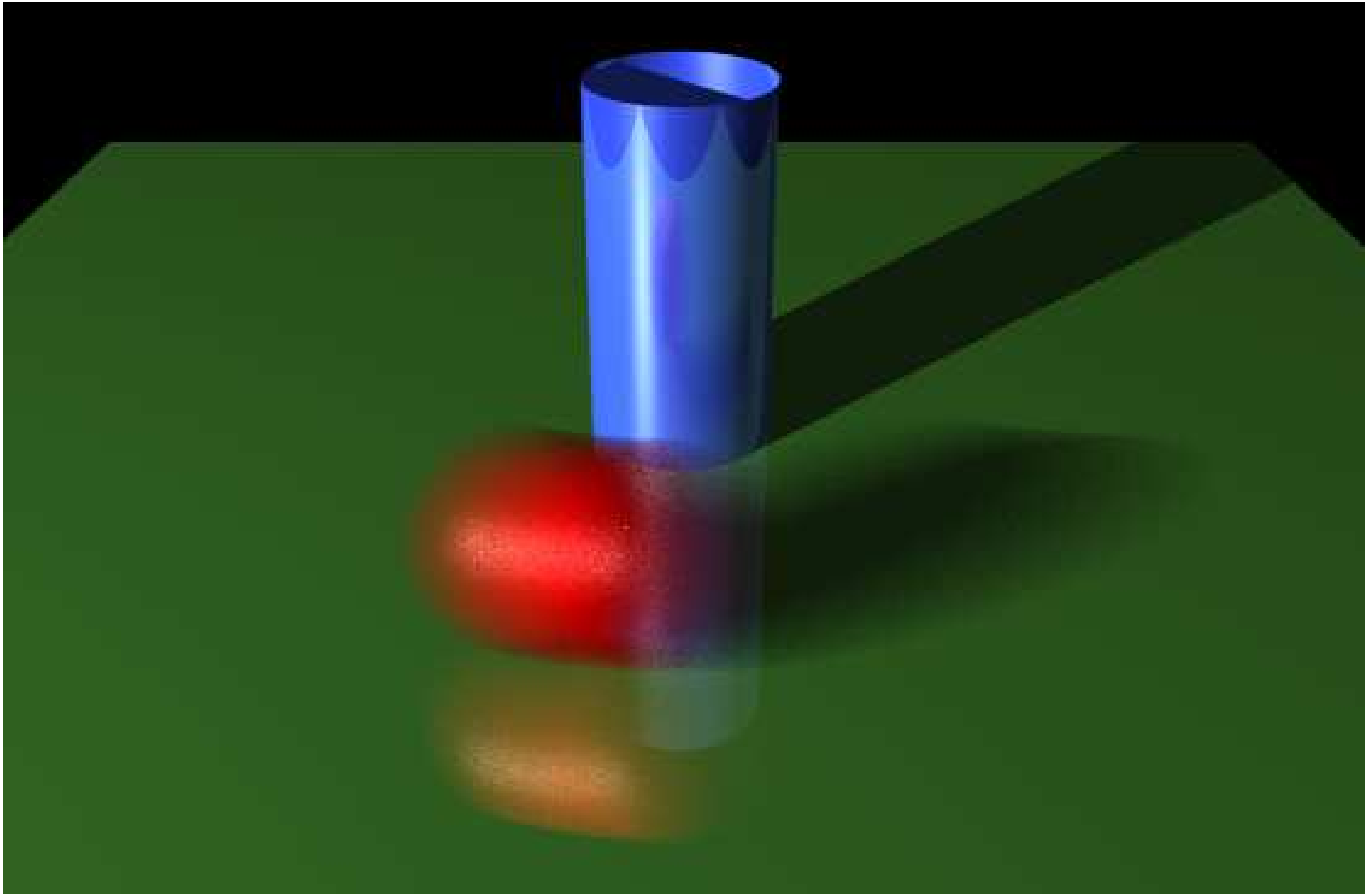
- Render time measurements
- Profiling
- Comparison with other renderers

Work so far

The following parts are implemented so far:

- Basic ray tracing algorithms, support for spheres, cylinders, planes, and polygon meshes.
- Scene graph with hierarchical transformations and key-frame animation for objects, lights and cameras (scene loading not implemented yet).
- Multithreading.
- Simple phong shader.
- Octree space subdivision for fast ray-primitive intersections.
- Soft shadows.
- Motion blur.

Work so far



Schedule

1/06 – 7/07	Finish the basic ray tracer and accompanying utilities (converters, exporters, etc)
7/07 – 15/08	Implement global illumination with photon mapping
15/08 – 29/08	Extensive testing, optimizations, and debugging period
29/08 – 18/09	Dissertation writeup

References

[1] J. T. Kajiya, "The rendering equation," in *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, (New York, NY, USA), pp. 143–150, ACM, 1986.

[2] R. L. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, (New York, NY, USA), pp. 137–145, ACM, 1984.

[3] H. W. Jensen, *Realistic Image Synthesis Using Photon Mapping*.
A. K. Peters, July 2001.