High Fidelity Radiosity Rendering at Interactive Rates

Steve Hardt*
hardts@mit.edu
MIT Laboratory for Computer Science

1 Overview

Existing radiosity rendering algorithms achieve interactivity or high fidelity, but not both. Most radiosity renderers optimize interactivity by converting to a polygonal representation and Gouraud interpolating shading samples, thus sacrificing visual fidelity. A few renderers achieve improved fidelity by performing a per-pixel irradiance "gather" operation, much as in ray-tracing. This approach does not achieve interactive frame rates on existing hardware.

Our research bridges the gap, by describing a data structure and algorithm which enable interactive, high-fidelity rendering of radiosity solutions. In essence, our algorithm "factors" the radiosity rendering computation into two components: an *offline* phase, in which a per-surface representation of irradiance is constructed; and an *online* phase, in which this representation is rapidly queried to produce a radiosity value at each pixel.

2 Offline Solution Phase

The key components of the offline phase are a heuristic discontinuity ranking algorithm, which identifies only the strongest discontinuities (shadow boundaries), and a hybrid quadtree-mesh data structure which prevents combinatorial interactions between most discontinuities. First, a quadtree-based radiosity solution is computed. Then, the quadtree leaves are triangulated according to the strongest discontinuities. And finally, a quadratic interpolant representing the irradiance function is fit to each triangle.

3 Online Rendering Phase

The online phase involves a novel use of perspective-correct texture-mapping hardware and parallel processors to produce accurate nonlinear, analytic shading effects.

We view the problem as: Given a list L of triangles, each annotated with an irradiance interpolant expressed in terms of triangle barycentric coordinates, assign each pixel P in the output image the radiosity value for that point B (on the visible triangle T) that projects to P. We do so with a multi-pass algorithm on a high-end graphics workstation.

(a) Generate an image $\mathcal{I}_1$ in which every pixel P contains an encoding of which triangle T is visible at P. This is accomplished with the polygon fill and depth-buffering hardware. Each triangle T is rendered as a solid "color", the "color" being a 24-bit encoding of the index of T in the list of triangles L. (Figure 1)

*Steve Hardt is advised by Prof. Seth Teller. Address: Room 226, 545 Technology Square, Cambridge MA 02139. Web: http://graphics.lcs.mit.edu/~hardts

(b) Generate an image $\mathcal{I}_2$ in which every pixel P contains an encoding of the barycentric coordinates B of the object-space point that projects to P. This is accomplished with the texture-mapping hardware. We create a single texture of barycentric coordinates, each texel (s,t) contains an encoding of s and t, and render each triangle with it. (Figure 2)

(c) Using $\mathcal{I}_1$ and $\mathcal{I}_2$, we generate the output image $\mathcal{O}$ by looping over, in parallel on multiple processors, all pixels P and evaluating T's irradiance interpolant at the barycentric coordinates B of the object-space point corresponding to P. (Figure 3)

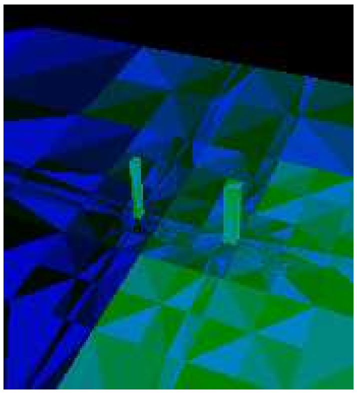

Figure 1: Triangle identifiers.

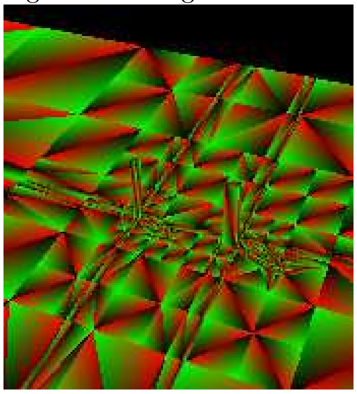

Figure 2: Barycentric coordinates.

Figure 3: Interpolant rendering.

References

[1] Daniel R. Baum, Holly E. Rushmeier, and James M. Winget. Improving radiosity solutions through the use of analytically determined form factors. *Computer Graphics (Proc. Siggraph '89)*, 23(3):325–334, 1989.

[2] M. F. Cohen and J. R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, 1993.

[3] Paul Heckbert. Discontinuity meshing for radiosity. *Third Eurographics Workshop on Rendering*, pages 203–226, May 1992.

[4] Seth Teller. Computing the antipenumbra cast by an area light source. *Computer Graphics (Proc. Siggraph '92)*, 26(2):139–148, 1992.

[5] Seth Teller and Pat Hanrahan. Global visibility algorithms for illumination computations. *Computer Graphics (Proc. Siggraph '93)*, 27:239–246, 1993.