

Radiosity on the GPU

Steve Crowe, Adam Micciulla

1 Introduction

Ever since its proposal, Radiosity has been a widely used method in Global Illumination rendering. Considering the parallel nature of the problem and the recent advances in GPU technology, it seems that implementing a Radiosity solver on the GPU would be very beneficial.

Our project will be based on a 2004 research paper by Greg Coombe, Mark Harris, and Anselmo Lastra entitled "Radiosity on Graphics Hardware." Although other papers have been able to map parts of the radiosity solution to the GPU, the method proposed is believed to be the first to move all radiosity calculations to the GPU, and the first method suitable for user interaction.

2 Implementation

The overall implementation of this solution is very similar to the Progressive Refinement approach to radiosity. In a typical Progressive Refinement approach, radiosity calculations are performed on subdivisions of the original mesh, called elements. For the hardware implementation, we are considering the elements to be equivalent to texels. Every associated polygon has two textures: radiosity and residual energy. We must store these as 16 bit textures packed into a single 32 bit texture. A Shooter within the Progressive Refinement approach can be considered a polygon in this implementation.

1. Visibility between shooters and patches:
 - (a) Select some shooter
 - (b) Render the scene from this shooter's POV using a stereographic projection.
 - (c) Store this texture for use as an ID buffer.
 - (d) We can then render each polygon using an orthographic projection.
 - (e) Using a fragment program, we can determine where the current fragment would be rasterized if it were rendered from the current shooter's point of view
 - (f) By using this location as an index into the ID buffer, we can determine if this fragment is visible from the shooter.
2. Form factor calculations
 - (a) During the rendering of each polygon, if the current fragment is visible, we are able to calculate an approximation of the form factor.
 - (b) Output the final radiosity and residual energy values.
3. Next shooter selection
 - (a) By examining the lowest level mipmap (1x1) for each polygon's texture, we can easily determine the polygon with the highest residual energy.

3 Deliverables

We intend to implement the design discussed in this paper using Cg to write fragment and vertex programs for NVidia hardware, specifically the NVidia GeForce 6000 series.

We will conduct more detailed benchmarks to see how quickly it could perform at different levels of complexity, and determine whether it would be applicable to a real-time environment where other GPU-heavy and CPU-heavy calculations are being performed, for example 3-D games or simulations.