Eurographics Symposium on Rendering (2004)
A. Keller, H. W. Jensen (Editors)

Combining Higher-Order Wavelets and Discontinuity Meshing: a Compact Representation for Radiosity

N. Holzschuch[1] and L. Alonso[2]

[1] ARTIS[†]GRAVIR-IMAG - INRIA [2] ISA/LORIA[‡] - INRIA

Abstract

The radiosity method is used for global illumination simulation in diffuse scenes, or as an intermediate step in other methods. Radiosity computations using Higher-Order wavelets achieve a compact representation of the illumination on many parts of the scene, but are more expensive near discontinuities, such as shadow boundaries. Other methods use a mesh, based on the set of discontinuities of the illumination function. The complexity of this set of discontinuities has so far proven prohibitive for large scenes, mostly because of the difficulty to robustly manage a geometrically complex set of triangles. In this paper, we present a method for computing radiosity that uses higher-order wavelet functions as a basis, and introduces discontinuities only when they simplify the resulting mesh. The result is displayed directly, without post-processing.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: I.3.5 [Computational Geometry and Object Modeling]:

1. Introduction

The radiosity method is a finite element method used for simulating light exchanges between diffuse surfaces. As such, it is used either for computing global illumination in diffuse scenes or as an intermediate step in other global illumination methods. Although other rendering methods, such as Bi-Directional Path Tracing or Photon Mapping are highly popular because they account for light exchanges between specular surfaces, many people still use radiosity because it offers the possibility to move the viewpoint in real-time after illumination computations.

However, radiosity methods are difficult to manage. The quality of the output is not always visually correct, and the memory cost of the algorithm can be quite high, since it needs to store a complete representation of the illumination on all objects in the scene. Hierarchical methods are used nowadays to reduce storage costs and computation time, and among them wavelet methods have proven interesting. Using higher-order wavelets as the basis functions, it is possible to approximate smoothly varying illumination with a small number of patches, reducing the memory cost.

A constant problem with basic radiosity methods is their misbehaviour near shadow boundaries. As most of these methods use an axis-aligned hierarchical grid for their finite-element computations, they are missing discontinuities that are not aligned with the grid. Solving this problem requires either using a finer grid size near shadow boundaries or using finite elements aligned with the shadow boundaries, called *discontinuity meshing*. The former method increases the memory cost, while the latter provides good quality reconstruction of illumination but the discontinuities are complex, and managing them in a robust and efficient way is still a research problem.

Since most higher-order wavelets are defined as tensor-products of 1D basis functions, they are only properly defined over parallelogram patches. As a consequence, they are in theory incompatible with discontinuity meshing, which produces complex polygons.

In this paper, we present an algorithm that combines higher-order wavelets with discontinuity meshing. We use

[†] ARTIS is a research project in the GRAVIR/IMAG laboratory, a joint unit of CNRS, INPG, INRIA and UJF

[‡] LORIA is a joint unit of CNRS, INPL, INRIA, UHP and Université Nancy 2.

wavelets, defined on a regular subdivision in places where they provide a good approximation, and we introduce discontinuities only in places where they reduce the complexity of the mesh. This selection of effective discontinuities is done during the refinement process, by the refinement oracle. The mesh produced is still a regular grid, but some of its patches are cut by discontinuities.

We use a fragment program to display quadric wavelets directly. We are displaying the results of our illumination computations immediately, without post-processing or final gather. We are exploiting the fact that higher-order wavelets with the proper refinement oracle result in apparently continuous functions after reconstruction, even in the absence of a specific step to enforce this continuity.

This paper is organised as follows: in the following section, we will review previous work on hierarchical – or wavelet – radiosity and discontinuity meshing, as well as integrating them. Then, in section 3 we will present our algorithm, and in section 4 we will present results and pictures from our experimentations. Finally, we will conclude and expose future research directions.

2. Previous Work

The radiosity method was first introduced for global illumination simulations by Goral *et al.* in 1984 [GTGB84]. It uses a finite element formulation of the rendering equation [KH84] for diffuse scenes, and gets a complete representation of global illumination. The radiosity method was later extended using a hierarchical formulation of the finite element method [HS92, HSA91]. The hierarchical representation limits the complexity of the radiosity algorithm to $O(n)$ instead of $O(n^2)$. This hierarchical formulation was later extended using a wavelet framework [GSCH93, SGCH93].

It is possible to use wavelets of different order (piecewise-constant basis, piecewise-linear basis, piecewise-polynomial basis). Early implementations of higher-order wavelets proved inefficient [WH97], until a complete analysis of the wavelet radiosity algorithm [CAH00] showed that with the right implementation, a good refinement oracle [BW96] and efficient memory management [SSSS98] they were actually more interesting than hierarchical piecewise-constant basis functions for global illumination simulations, with smaller memory costs and shorter computation times.

Piecewise polynomial wavelets are more costly for each patch of the finite element formulation, requiring $(k+1)^2$ coefficients for a wavelet basis made of polynomials of degree k. But they provide a better approximation of the illumination function, resulting in a smaller number of patches. The study by Cuny *et al.* [CAH00] showed that most of the time, the reduction in the number of elements more than compensates for the extra cost for each element, allowing a faster computation of radiosity and a smaller memory cost.

However, many scenes on which we wish to compute global illumination exhibit sharp discontinuities of the illumination functions, for example shadows caused by point light sources or small area light sources, or shadows caused by occluders that are close to the receiver. Regular hierarchical basis of continuous polynomials are unable to model such discontinuities. In the presence of these discontinuities, most radiosity algorithms refine the hierarchy a lot, using very small patches to approximate the radiosity function. The result is that the number of patches used near the discontinuity is roughly independent from the order of the basis function. Since each patch stores $(k+1)^2$ coefficients, wavelet bases of higher-order polynomials end up being more costly at these discontinuities.

Discontinuities of the radiosity function can be computed using geometrical methods [LTG92, Hec92] [DF94, SG94, GS96, DDP02]. An adaptive mesh based on these discontinuities provides a better approximation of the radiosity function [LTG92, Hec92]. Radiosity methods based on the discontinuity mesh have been proposed, either with classical radiosity [LTG92, Hec92, Stu94, DF94] or with hierarchical radiosity [LTG93, DS96, DDP99]. All these methods start with the complete set of discontinuities, triangulating it and refining it as necessary. The entire set of discontinuites is quite large, giving a very complex mesh as a starting point. Managing this mesh proves complicated, and the associated memory cost is not neglictible. [DS96] used a regular mesh for visible areas, but kept the triangulated set of discontinuities for penumbra regions.

Several of the discontinuities in the discontinuity mesh are not visible in the radiosity function. Simplifications of the discontinuity mesh have been suggested [DF94, HWP97, Hed98]. But as they are computing discontinuites before the illumination computations, this selection uses only geometrical tools and has not access to illumination information.

In our algorithm, however, we use a regular subdivision as often as possible, and we only introduce discontinuities if they result in a simpler mesh. Significative discontinuities are thus naturally selected during the hierarchical refinement process.

A single paper has used Discontinuity Meshing together with wavelet radiosity with higher order-basis functions [PB95]. Their study is quite complete, but they used only very simple scenes for their tests: a single patch with a single discontinuity. As a consequence, they could not identify several problems that only occur in larger scenes, such as intersecting discontinuities or the cost of computing push-pull coefficients: their method would not scale to scenes much bigger. They tried to merge wavelets with discontinuities by finding a wavelet-compatible parametrization of the patch that followed the discontinuity. This causes a complex computation of push-pull coefficients for each hierarchical level. Also, building such a parametrization is not always possible, in the case of intersecting discontinuities. Finally, their approach does not address the problem of managing the set of discontinuities. Our algorithm, by contrast, keeps the same parametrization for all patches in the hierarchy, making it easy to compute push-pull coefficients. We can deal with multiple discontinuities and intersecting discontinuities. Each discontinuity inserted in the hierarchy is treated only at its hierarchical level.

3. Algorithm

In this section, we present our algorithm for merging radiosity using higher-order wavelets with meshing discontinuities. We start with a short summary of the Hierarchical Radiosity algorithm, and how it has been adapted to higher-order wavelets (section 3.1). Then we present our algorithm for merging the wavelet bases with discontinuities (section 3.2). Some finer points of the implementation are explained in section 3.3.

3.1. Wavelet Radiosity

3.1.1. The Hierarchical Radiosity Algorithm

In Wavelet Radiosity, each surface of the scene carries a hierarchical representation of illumination, using the wavelet basis. This representation is computed iteratively, through three essential steps:

- refinement of interactions,
- propagation of energy,
- push-pull.

At the beginning of the algorithm, we select the surface with the largest unshot energy, and indentify all surfaces that are potentially visible from it. We establish interactions between the shooting surface and all these receiving surfaces.

We then *refine* these interactions, in a hierarchical manner. At each point in time, we consider the current multi-scale representation of the interaction, and check whether it is accurate enough, according to the *refinement oracle*. If not, we refine the interaction, by subdividing either the shooting surface or the receiver.

Once we are satisfied with the level of precision on the interaction, we *propagate* the energy by sending the unshot energy of the shooting surface to the receivers, updating the wavelet coefficients on the receivers.

After these steps, the unshot energy of the shooting surface is set to zero, and we pick the surface with the largest unshot energy as the next shooting surface.

After the propagation, the different levels of the hierarchy on each surface have received energy, but there isn't a consistent representation of the energy received at all hierarchical levels. This representation must be reconstructed before we can use the hierarchical representation for shooting or for display. It is done during the *push-pull* step. The *push-pull* step is a recursive procedure, where parent nodes add their energy to their children, and the children's energy is collected in each parent and averaged.

3.1.2. Using Higher-Order Wavelets

Using higher-order wavelets, such as Multi-Wavelets ($\mathcal{M}_2$ and $\mathcal{M}_3$) [Alp93], does not change the algorithm, except in these details:

- each patch carries a wavelet representation of the radiosity function. The $\mathcal{M}_n$ basis is made of polynomials of degree $n-1$, so each patch has n^2 basis functions and stores n^2 coefficients.
- The interaction between two patches implies computing the influence that each wavelet coefficient on the shooting patch has on every wavelet coefficient on the receiving patch. Each of these influence coefficients is expressed as an integral, which is approximated using quadratures. As there are n^2 coefficients on each patch, we must evaluate n^4 integrals.
- The push-pull step implies computing the influence that each wavelet coefficient on the parent patch has on every wavelet coefficient on the children patches, and reciprocally. These influences are also expressed as integrals. These integrals only depend on the respective geometry of the parents and children patches in the hierarchy. For a regular subdivision, the push-pull coefficients are therefore constant on the hierarchy, and are pre-computed. For irregular subdivision, the push-pull coefficients must be recomputed at each level, a potentially costly step.
- As 2D wavelets are usually defined as tensor-products of 1D wavelets, they are only defined over a parallelogram. Researchs have shown how to extend this definition for complex planar surfaces [HCA00] and for parametric curved surfaces [ACP*01].
- Given the large number of coefficients for each interaction (n^4, as much as 81 coefficients for polynomials of degree 2), it is important to avoid storing them. Once we have treated an interaction, we delete all its coefficients. This strategy can result in computing the same interaction coefficients twice, but the gain in memory largely offsets the potential loss in time [SSSS98, CAH00].

3.2. Combining Wavelets and Discontinuity Meshing

3.2.1. The algorithm

Our algorithm works as follows:

- For each shooting surface, for each receiving surface, we compute the set of discontinuities on the receiving surface.
- We proceed with the usual refinement of interaction, using the oracle and a regular subdivision.
- When the refinement oracle identifies that the interaction should be subdivided only because of a discontinuity, it

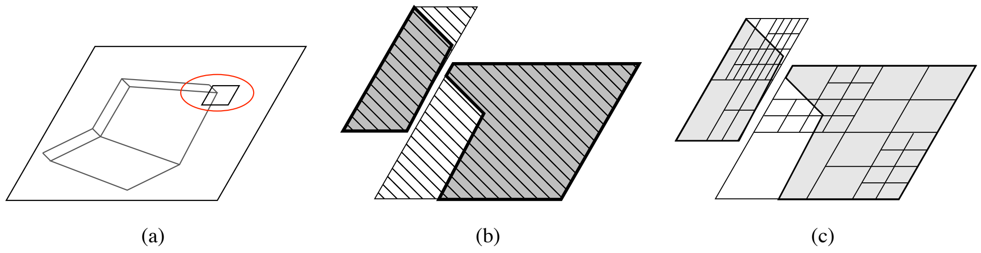


Figure 1: *A patch cut by a discontinuity (a) results in two children patches. For each children patch, we identify the enclosing parallelogram (b). We conduct standard wavelet radiosity on each parallelogram (c).*

introduces a *discontinuity-based subdivision* instead of a regular subdivision.

- Discontinuity-based subdivision works by:
  - Computing the intersection of the current patch with the discontinuity.
  - For each part of the subdivided patch, identify the smallest parallelogram that encloses it.
  - Apply our radiosity algorithm using a regular subdivision over each parallelogram (see Figure 1).
- Once we are satisfied with the level of refinement for this interaction, we propagate the energy, then erase the discontinuities and the interaction coefficients. Discontinuities that have not been used for subdivision are forgotten.

We want to use the regular subdivision as much as possible for its robustness and simplicity. Our algorithm only introduces discontinuities if they are considered important by the refinement oracle. Smooth transitions that can be properly approximated by the wavelet basis will not be introduced in the hierarchy.

In the following paragraphs, we review each step of this algorithm in detail: refinement oracle, discontinuity-based subdivisions, push-pull over a discontinuity, intersection of discontinuities.

3.2.2. Refinement oracle and selection of discontinuities

We use the refinement oracle described in previous publications [BW96, CAH00]: for each patch, we select testing points, where we compute radiosity directly. The values computed are compared with values obtained using the wavelet basis. If the norm of the differences is above the refinement threshold, the oracle concludes that we should refine.

This oracle works well, especially if the testing points are chosen with a good heuristics. By putting some of the testing points on the boundaries of the patches, we have found that we obtain a representation of radiosity that looks continuous without having to ensure this continuity in post-processing (see [CAH00] and Figure 2 for an example using $\mathcal{M}_3$ wavelets).

In our algorithm, we do two computations of the refinement oracle: one with standard visibility computations, and one assuming full visibility. If their results differ, visibility is the only reason for subdivision and we introduce a discontinuity-based subdivision.

Subdivisions are thus only introduced in the hierarchy if they actually cancel further refinements on at least one side, resulting in a more compact hierarchy. For point light sources, introducing a subdivision generates a coarse mesh on both sides of the subdivision (see Figure 2(a)). For area light sources, introducing subdivisions creates a coarse mesh in fully lit areas and in the umbra, while the penumbra is more refined (see Figure 2(b)).

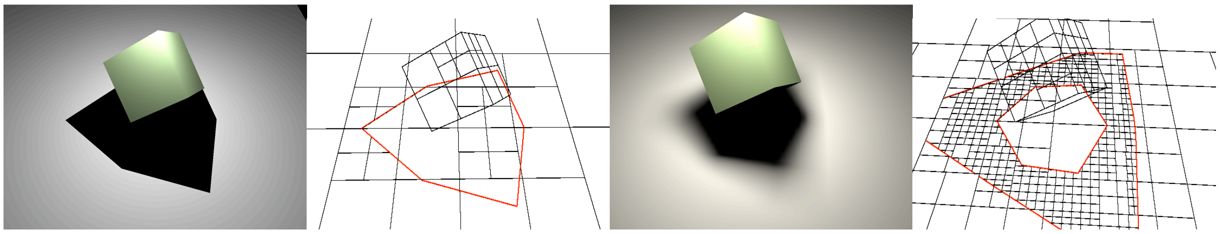
For stability and robustness, a discontinuity is introduced only if the intersection between the discontinuity and the current patch is simple enough. Thus our algorithm only has to manage simple patches and surfaces. For complex occluders casting a combination of simple and complex discontinuities, only the simple discontinuities are introduced in the mesh (see Figure 11).

In our implementation, we have used the following criteria for selecting simple discontinuities: at least one of the patches resulting from the discontinuity-based refinement must be convex, and the number of vertices in each polygon remains below a certain threshold.

3.2.3. Discontinuity-based subdivisions

Once we have selected a patch for discontinuity-based subdivision, we compute the intersection between the patch and the discontinuities, resulting in two separate patches. Most of the time, these sub-patches are neither parallelograms nor triangles. For each of the sub-patches, we build the smallest enclosing parallelogram (see Figure 1). We then use these enclosing parallelograms instead of the patches in the radiosity algorithm as we would use standard patches:

- For *radiosity reception*, the enclosing parallelogram is



(a) Point light source (b) Area light source

Figure 2: $\mathcal{M}_3$ *(quadric) wavelets with discontinuity meshing on simple scenes*

treated as a standard receiver. It is subdivided normally, using regular subdivision.

- For *radiosity emission*, only the actual sub-patch is allowed to emit radiosity; other parts of the enclosing parallelogram are not allowed to emit. Following previous research [HCA00] we do this through the quadrature weights, during the computation of Gaussian quadratures. We see each quadrature weight as the representative of an *area of influence* for the quadrature point (see Figure 3). We modulate the quadrature weight by the percentage of this area of influence that is inside the actual sub-patch.
- For *push-pull*, we use the standard push-pull coefficients since we have a standard subdivision.

3.2.4. Push-Pull Coefficients over a discontinuity

On most steps of the radiosity algorithm, our method uses classical methods. The main difference lies in the push-pull step over the discontinuity.

The enclosing parallelograms of the children patches are overlapping, and we need the push-pull step to compensate for this. Let us assume a patch p has been subdivided into two children patches p_1 and p_2. The children patches p_i are enclosed into parallelograms e_i. Each of the patches have its own set of wavelet basis functions: ϕ_j on p, ϕ^i_j on e_i. The radiosity function is expressed as:

$$\begin{aligned} B_p(x) &= \sum_j \alpha_j \phi_j(x) \\ B_{e_i}(x) &= \sum_j \alpha^i_j \phi^i_j(x) \end{aligned}$$

3.2.4.1. Push Coefficients: For the push step, we need to project B_p on the basis functions of the children e_i. The wavelets coefficients of the projection will be added to the wavelet coefficients on each child e_i. Since, on each patch, wavelets functions form an orthonormal basis, wavelet coefficients are expressed as the scalar product of the radiosity function with the basis functions:

$$\alpha^i_j = \langle B_{e_i} | \phi^i_j \rangle_i$$

where the subscript i on the dot product expresses the fact that the integration takes place on e_i. We are looking for the contribution of B_p to the α^i_j, push^i_j:

$$\begin{aligned} \text{push}^i_j &= \langle B_p | \phi^i_j \rangle_i \\ &= \langle \sum_k \alpha_k \phi_k | \phi^i_j \rangle_i \\ &= \sum_k \alpha_k \langle \phi_k | \phi^i_j \rangle_i \\ &= \sum_k \alpha_k C^i_{kj} \end{aligned}$$

The push coefficients, C^i_{kj}, only depend on the basis functions and on the relative geometry of p and e_i. We have an integral expression for the push coefficients:

$$C^i_{kj} = \langle \phi_k | \phi^i_j \rangle_i = \int_{e_i} \phi_k(x) \phi^i_j(x) dx$$

3.2.4.2. Pull coefficients: For the pull step, we need to combine together the radiosity functions on patches p_i, and express this radiosity on the wavelet basis for patch p. As the e_i patches are overlapping, we restrict the definition of B_{e_i} to its support. We use the characteristic function of e_i, δ_{e_i}, defined as being equal to 1 on e_i and 0 everywhere else.

Combining together the radiosity functions computed on the children gives us:

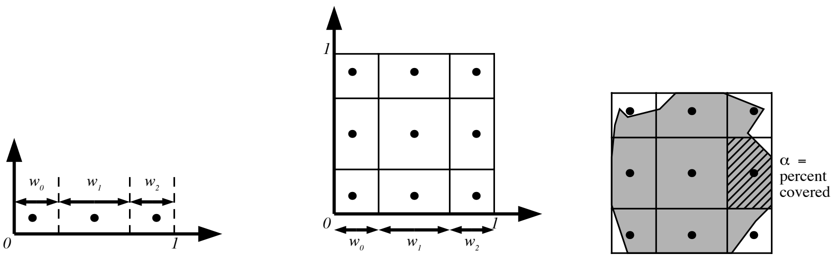
$$B_{e_1}(x)\delta_{e_1}(x) + B_{e_2}(x)\delta_{e_2}(x) = \sum_i \sum_j \alpha^i_j \phi^i_j(x) \delta_{e_i}(x)$$

The pull step projects this combined function on the wavelet basis for p:

$$\begin{aligned} \text{pull}_k &= \sum_i \sum_j \alpha^i_j \langle \phi^i_j \delta_{e_i} | \phi_k \rangle \\ &= \sum_i \sum_j \alpha^i_j D^i_{jk} \end{aligned}$$

The pull coefficients, D^i_{jk} depend on the geometry of the subdivision:

$$D^i_{jk} = \langle \phi^i_j \delta_{e_i} | \phi_k \rangle = \int_p \phi^i_j(x) \delta_{e_i}(x) \phi_k(x) dx$$



(a) on the unit segment (b) on the unit square (c) on the extended domain

Figure 3: *The weights of the quadrature points can be seen as the area of a* zone of influence.

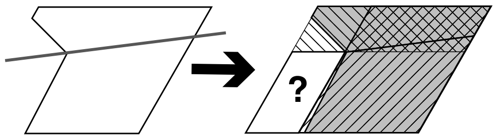


Figure 4: *A non-convex patch cut along a discontinuity can result in two children whose enclosing parallelograms do not cover the enclosing parallelogram of the parent.*

3.2.4.3. Computation of push-pull coefficients For the push-pull step over the discontinuity, we have an integral expression, which we approximate using Gaussian quadratures. As we are integrating a discontinuous function (δ_{e_k}), we might have accuracy problems in the computation. We compensate by using a large number of sampling points. Also, once we have computed the coefficients for one patch, we check that they are consistent with each other, and that there is no creation or destruction of energy during the push-pull step. Should we detect an inconsistency, we recompute them with more precision.

Push-pull coefficients are then stored on the hierarchy. Because these special push-pull coefficients only happen once for each discontinuity-based subdivision, we can afford to spend some time computing them.

3.2.5. Intersection of several discontinuities

The patches resulting from a discontinuity-based subdivision are not necessarily convex. If a non-convex patch is cut by another subdivision, the enclosing parallelograms of the children do not cover the enclosing parallelogram of the parent patch (see Figure 4).

This configuration appears when discontinuities from two light sources intersect each other, or when the umbra and penumbra boundaries touch each other, for an occluder that is in contact with the receiver.

When it appears, it causes the push-pull coefficients to be incomplete in their definition. To account for this, we extend the enclosing parallelograms of the children so that their union covers the enclosing parallelogram of the parent patch. Except for this small point there is no special case in our algorithm for dealing with several light sources and intersecting discontinuities (see Figure 5).

3.3. Implementation details

In this section, we review implementation details of our algorithm. The points described here are not *essential* to our algorithm; others could use different approaches, *e.g.* for computing discontinuities, or for handling visibility queries in radiosity computations. However, the approach we used to solve these problems can be interesting to other researchers.

We have used non-conventional solutions for computing discontinuities, in the refinement oracle, for visibility queries and for displaying results:

Finding Discontinuities: We only need the set of discontinuities for the interaction currently being refined. We compute extremal discontinuities (umbra and penumbra boundaries), using a method based on the `GLU Tesselator` [SWND03]. Our method identifies EV, VE and EEE events, converts these events into 2D polygons corresponding to their intersection with the plane of the receiver, then uses the `GLU Tesselator` to compute the union and the intersection of these 2D polygons. Our algorithm for finding discontinuities is not complete (it can miss some discontinuities) but it is robust and it finds the most important discontinuities.
Umbra and penumbra boundaries are not necessarily linear: on EEE evetns, parts of them can be conic curves. Our algorithm deals with such conics in a straightforward manner.
Once we have computed the umbra and penumbra contours, we have to answer *position queries*: "is this point inside the umbra or not?". We store the contours in an arrangement of line segments, using trapezoidal maps(see,

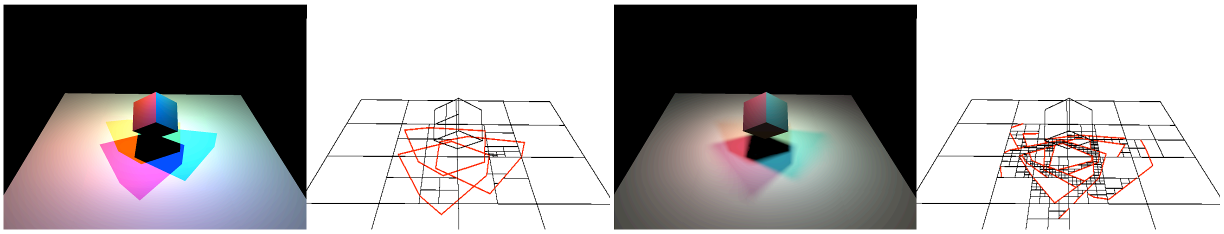

(a) Point light sources ($\mathcal{M}_3$ wavelets) (b) Area light sources ($\mathcal{M}_3$ wavelets)

Figure 5: *Combining together several discontinuities (both scenes have three light sources, red, green and blue, located in a triangle above the cube).*

e.g. [BDS*92, CGA]). This randomized data structure answers our positions query in average time $O(\log n)$, with creation time $O(n \log n)$ and memory cost $O(n)$.

Handling Discontinuites in the Refinement Oracle: The refinement oracle takes sampling points on the receiving patch. Some sampling points can lie on a discontinuity, which makes their exact value unknown. To avoid unnecessary refinement, points lying on a discontinuity can take a different value in the oracle depending on the patch being considered.

Handling Visibility Queries: In our radiosity computations, we need the percentage of the light source that is visible from the receiving points. Previous implementations used a geometric data-structure, the *Backprojection* [DF94] to compute an exact value of this percentage. We are computing it instead using an OpenGL extension, `OcclusionQuery` [ARB], which gives us the percentage of the pixels of the light source that are visible from the receiving point. In our experiments, occlusion queries are more robust than the geometric data structure while having the same speed, and they are much faster than casting rays, while giving more precise results.

Displaying Results: $\mathcal{M}_3$ wavelets give quadrically varying functions; they are displayed using a small fragment program (10 lines of code). Linear interpolation from the graphics hardware (*Gouraud shading*) is not perfect for $\mathcal{M}_2$ wavelets, which are bilinear functions. It is possible to replace this linear interpolation by a small fragment program.

4. Experimentations and Results

4.1. Experimentation protocol

Test scenes: We have used two different test scenes: the Cabin, from Radiance set of test scenes, and Room 523 from the Soda Hall model. For each scene, we used either point light sources or area light sources, giving a total of four test scenes. On all test scenes, we computed direct and indirect illumination. Pictures of the test scenes are available in Figures 7 and 13 (see color plates).

Wavelet Bases: We have tested our algorithm with the first three multi-wavelets bases: $\mathcal{M}_1$ (Haar), $\mathcal{M}_2$ (piecewise-linear) and $\mathcal{M}_3$ (piecewise-quadric). In the pictures, Haar wavelets are displayed after a post-processing step to ensure continuity, $\mathcal{M}_2$ wavelets are displayed using standard linear interpolation from the graphics hardware and $\mathcal{M}_3$ wavelets use a fragment program for the quadrically varying part.

Material: All computations were done on the same computer: a 2.4 GHz Pentium IV, with 1Gb memory and an NVIDIA GeForce FX 5600.

4.2. Visual comparison for point light sources

The first reason to use discontinuity meshing is the quality of the illumination computed. Adapting the mesh to the discontinuities produces a radiosity function that looks pleasing to the eye.

The leftmost columns of Figures 6 and 12 show a side-by-side comparison of the different wavelet bases on a specific detail of the Cabin test scene, with a point light source. All pictures were generated with the same computation time (25 s) to give a fair comparison of the different wavelet bases. Without discontinuity meshing, the most satisfying representation is obtained with $\mathcal{M}_2$ wavelets, but artefacts are clearly visible along the discontinuity line; Haar and $\mathcal{M}_3$ wavelets are visually not acceptable within the prescribed time frame; they would eventually achieve a satisfying result, but for a longer computation time.

With discontinuity meshing, all wavelets bases achieve a visually pleasing result. Our algorithm for merging discontinuities with wavelets thus achieves a visually better result in the same computation time.

We did a similar comparison for Room 523 of the Soda Hall. Figure 8 shows the pictures obtained with the different wavelet bases on a detail of the room. All pictures were

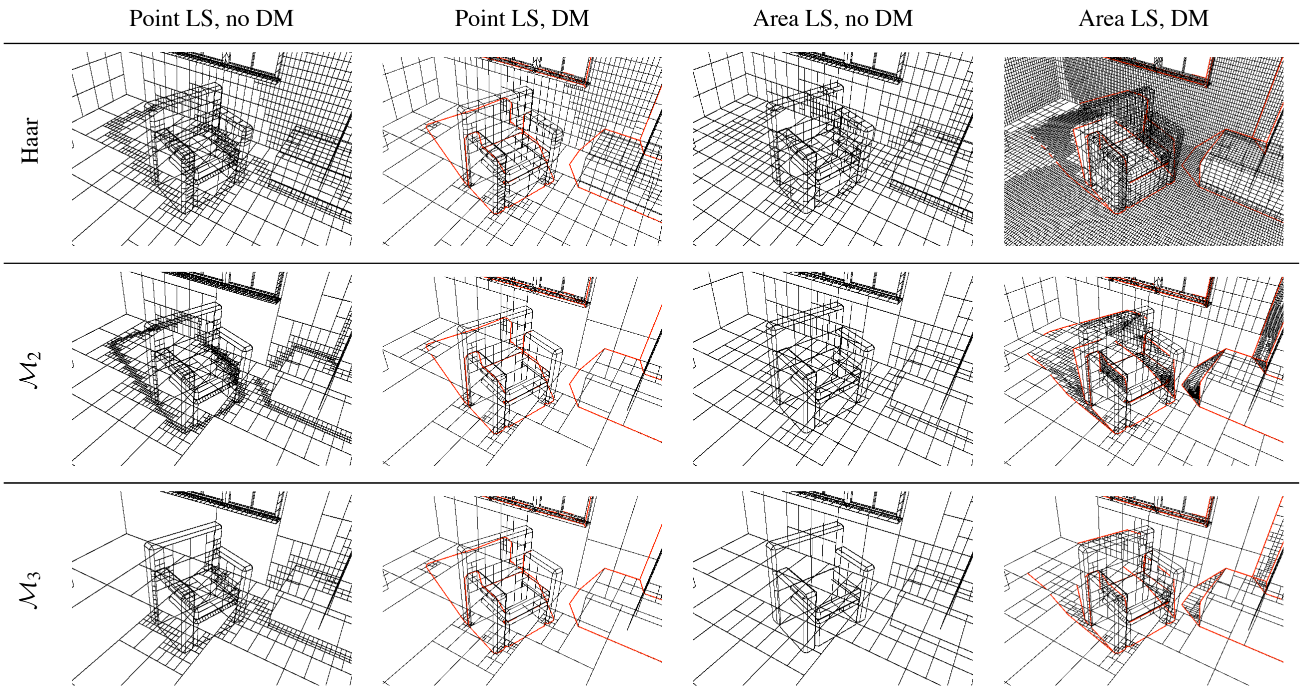


Figure 6: *Wireframe version of simulation for the Cabin test scene. See also the color plates.*

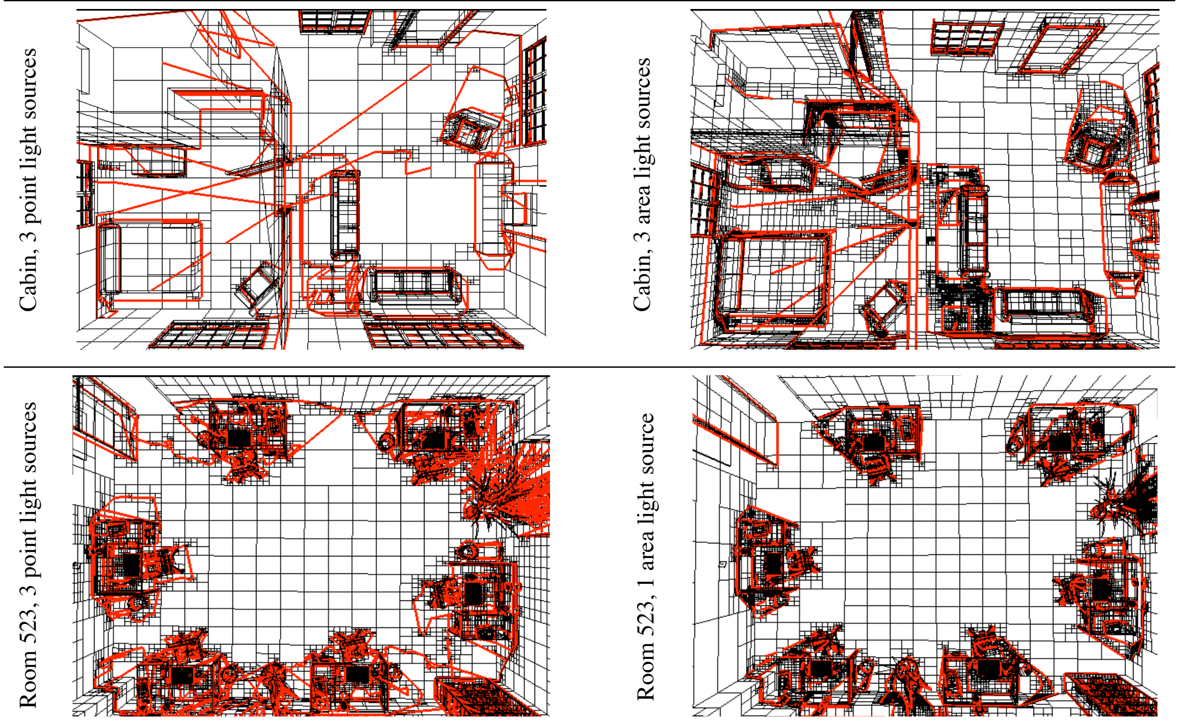
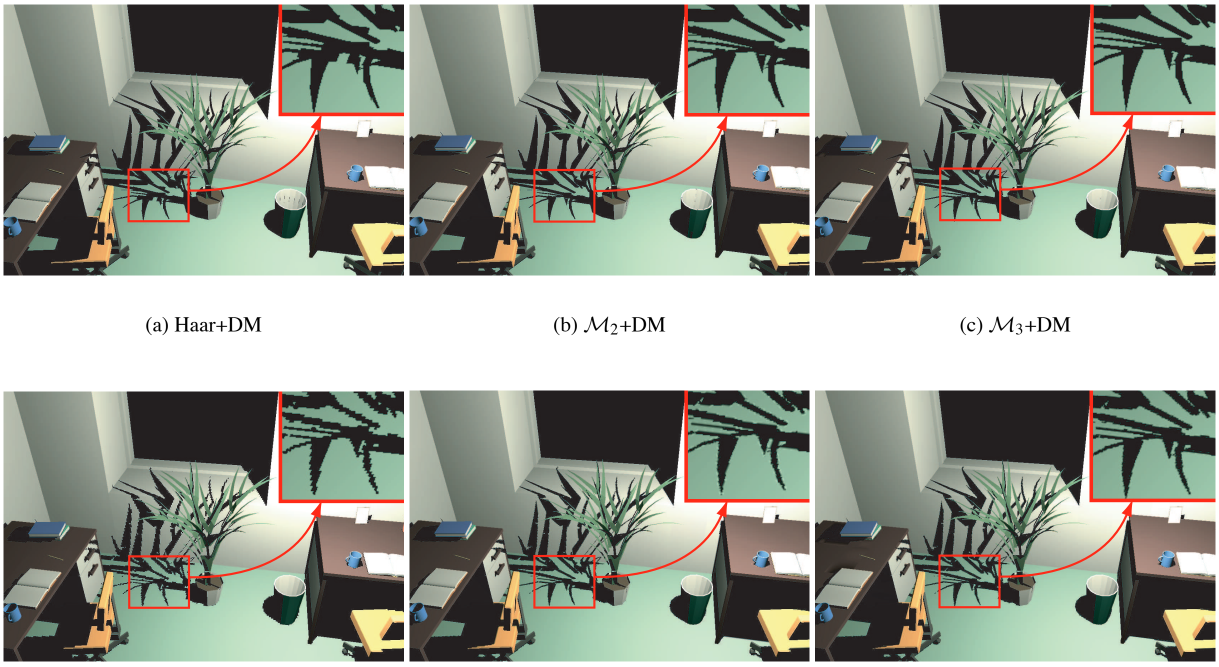

Cabin, 3 point light sources
Cabin, 3 area light sources
Room 523, 3 point light sources
Room 523, 1 area light source

Figure 7: *Wireframe version of our test scenes after simulation with $\mathcal{M}_3$ wavelets. See also the color plates.*



(a) Haar+DM (b) $\mathcal{M}_2$+DM (c) $\mathcal{M}_3$+DM

(d) Haar (e) $\mathcal{M}_2$ (f) $\mathcal{M}_3$

Figure 8: *Visual comparison of the different wavelet bases for a point light source. All pictures used roughly 190 s computation time.*

generated with approximately the same computation time (190 s).

From a distant point of view, all the pictures are of comparable quality. On a large scene like this, with a high number of discontinuities, the time spent estimating the discontinuities and dealing with them becomes equivalent to the time it takes to do regular subdivisions.

However, when we look closely at the shadow boundaries, ringing artefacts and staircase effects become clearly visible (see the full resolution inserts).

We also compared the memory costs for both versions of the program (with discontinuity-based subdivision and without). Figure 9 shows the memory costs for all three wavelet bases, for the pictures on Figures 6, 12 and 8. On a scene with a large number of discontinuities, such as Room 523, our algorithm results in an important gain in memory costs. Each discontinuity introduced replaces a large number of regular patches, resulting in a net gain. On the Cabin test scene, with the prescribed time limit, the refinement was not pushed to the same levels. As a consequence, the memory gain is not as strong.

In short, our algorithm for merging discontinuities with higher-order wavelet bases always gives better results than existing algorithms, with a smaller memory cost.

4.3. Visual Comparison for Area Light Sources

The rightmost columns of Figures 6 and 12 show the same comparison of the different wavelet bases for the same detail of the Cabin test scene, this time using an area light source. All pictures were generated with approximately the same computation time (240 s). This time, the benefits of using the discontinuity-based approach appear very clearly. All three wavelet bases greatly outperform the non-discontinuity-based versions. This is because computing the discontinuities speeds-up the visibility computations, the most expensive step in hierarchical radiosity. Within discontinuity-based wavelet methods, $\mathcal{M}_2$ and $\mathcal{M}_3$ wavelets produce the nicest result.

For comparison, Figure 10 shows the memory costs for all wavelet bases on this test scene. Notice that this time, the memory cost is bigger *with* discontinuity meshing than without.

This is a side effect of our comparison method: the time given for the simulation was much too short for the system without discontinuities. It has just computed a crude ver-

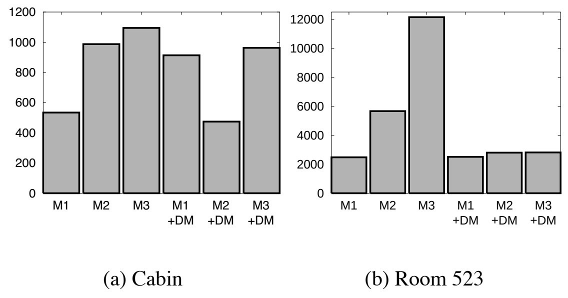


Figure 9: *Memory costs (in Kb) for simulations on our test scenes, with point light sources.*

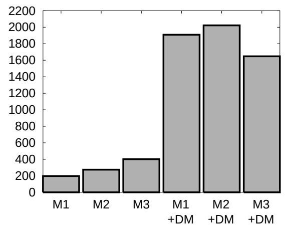


Figure 10: *Memory costs (in Kb) for simulations on the Cabin test scene, with an area light source.*

sion of illumination. If we give it more time for simulation, it eventually computes a nice version of the illumination, at a higher memory cost. Note that the memory cost without discontinuities increases with the order of the wavelet base. This is consistent with previous research [HCA00]: for crude estimates of the illumination, the memory cost increases with the order of the wavelet base, while for high-quality estimates, the memory cost decreases with the order of the wavelet base.

4.4. Selective choice of discontinuities

In places where the radiosity is smoothly varying, our algorithm can choose not to introduce discontinuities, keeping the regular subdivision. This effect appears clearly in the wireframe representations of our test scenes (see the right-most column of Figures 6 and 12, and Figure 7).

Figure 11 also shows a detail of the Room 523 test scene (with an area light source) where complex discontinuities exist, but were not introduced in the mesh. This behaviour is more likely to occur with area light sources, which produce smoothly varying illumination, than with point light sources, where there is always a C^0 discontinuity at a shadow boundary.

4.5. Influence of the minimal area parameter

An important issue in the practical use of radiosity algorithms is the choice of parameters. A bad choice of the parameters results in a long computation time or a simulation that is not visually pleasing – sometimes both.

The minimal area for patches is one of these parameters. Without discontinuity-based refinement, this minimal area is reached for many patches on all sharp discontinuities (see the wireframe representations of the test scenes, in Figure 6 and additional materials). A variation of this parameter has important consequences on the computation time, on the memory cost and on the quality of the result. Dividing this minimal area by 2 results in twice as many patches being used to represent each sharp discontinuity, potentially doubling the computation times and memory cost.

With discontinuity-based refinement the minimal area is not reached, except in places where discontinuities are too complex. Hence, a variation of this parameter has little consequences on the computation times and memory costs. Our algorithm has almost cancelled the influence of the minimal area parameter. The user of our radiosity system has one main parameter, the maximum value of the error on each interaction. The minimal area still has an effect on the quality of the simulation, computation time and memory cost, but it is a minor effect.

5. Conclusion and Future Directions

In this paper, we have presented a new algorithm for radiosity computations, that combines higher-order wavelets with discontinuity meshing. Our algorithm uses regular subdivision for wavelets where it is practical, and switches to discontinuity-based subdivision where discontinuities exist. Only discontinuities that are important in the computation of the illumination solution are actually introduced in the mesh. This results in a compact representation of radiosity, with a good compromise between quality and cost.

This representation can be displayed directly on the screen, or it can be used as a starting point for more complete illumination computations, such as Monte-Carlo illumination.

Our algorithm is robust enough to handle complex discontinuities. It automatically discards discontinuities which are not important enough for the radiosity computations, providing a good way to manage the complicated set of discontinuities.

In future work, we want to combine our algorithm with a robust computation of visibility discontinuities (e.g. [DD02]). We also want to combine our work with separate works allowing higher-order wavelet radiosity computations on curved surfaces [ACP*01] and triangular meshes.

In a separate direction of research, although our algorithm

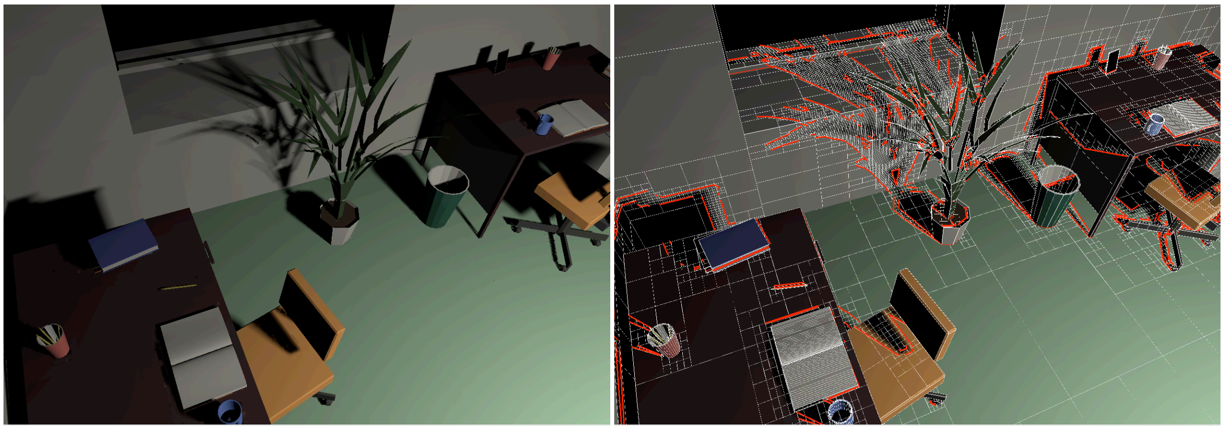


Figure 11: *Our algorithm only inserts discontinuities that are perceived as useful by the refinement oracle ($\mathcal{M}_3$ wavelets).*

only inserts discontinuities as they are needed in the refinement process, it starts by computing all potential discontinuities for the current interaction, a costly preliminary step. We will explore the possibility to suppress this step, using standard refinement but detecting inside the refinement oracle that subdivision is probably caused by a discontinuity, then only computing and inserting this discontinuity in the mesh. This would reduce the computation cost of our algorithm. In our experiments, almost all the discontinuities caused by indirect lighting are not important enough to justify their insertion in the hierarchy. It is therefore not practical to compute them in advance.

6. Acknowledgements

This work was partly funded by the "Région Rhône-Alpes" under the DEREVE project.

The Soda Hall model was created by the original Berkeley Walkthru team under the direction of Prof. Carlo H. Sequin. For more information on the Soda Hall, see `http://www.cs.berkeley.edu/~sequin/soda/soda.html`

References

[ACP*01] ALONSO L., CUNY F., PETITJEAN S., PAUL J.-C., LAZARD S., WIES E.: The virtual mesh: A geometric abstraction for efficiently computing radiosity. *ACM Transactions on Graphics 20*, 3 (July 2001), 169–201.

[Alp93] ALPERT B. K.: A class of bases in L^2 for the sparse representation of integral operators. *SIAM Journal on Mathematical Analysis 24*, 1 (1993), 246 – 262.

[ARB] ARB_occlusion_query. `http://oss.sgi.com/projects/ogl-sample/registry/ARB/occlusion_query.txt%`.

[BDS*92] BOISSONNAT J.-D., DEVILLERS O., SCHOTT R., TEILLAUD M., YVINEC M.: Applications of random sampling to on-line algorithms in computational geometry. *Discrete and Computational Geometry 8*, 1 (1992), 51–71.

[BW96] BEKAERT P., WILLEMS Y.: Error Control for Radiosity. In *Rendering Techniques '96 (7th Eurographics Workshop on Rendering)* (1996), pp. 153–164.

[CAH00] CUNY F., ALONSO L., HOLZSCHUCH N.: A novel approach makes higher order wavelets really efficient for radiosity. *Computer Graphics Forum (Eurographics 2000) 19*, 3 (2000), C–99–C–108.

[CGA] CGAL, Computational Geometry Algorithms Library. `http://www.cgal.org`.

[DD02] DUGUET F., DRETTAKIS G.: Robust epsilon visibility. *ACM Transactions on Graphics (SIGGRAPH 2002) 21*, 3 (2002).

[DDP99] DURAND F., DRETTAKIS G., PUECH C.: Fast and accurate hierarchical radiosity using global visibility. *ACM Transactions on Graphics 18*, 2 (1999), 128–170.

[DDP02] DURAND F., DRETTAKIS G., PUECH C.: The 3d visibility complex. *ACM Transactions on Graphics 21*, 2 (Apr. 2002).

[DF94] DRETTAKIS G., FIUME E.: A Fast Shadow Algorithm for Area Light Sources Using Backprojection. In *Computer Graphics Proceedings, Annual Conference Series, (ACM SIGGRAPH '94)* (1994), pp. 223–230.

[DS96] DRETTAKIS G., SILLION F.: Accurate Visibility and Meshing Calculations for Hierarchical Radiosity. In *Rendering Techniques '96 (7th Eurographics Workshop on Rendering)* (1996), pp. 269–278.

[GS96] GHALI S., STEWART A. J.: A Complete Treatment of D1 Discontinuities in a Discontinuity Mesh. In *Graphics Interface '96* (May 1996), pp. 122–131.

[GSCH93] GORTLER S. J., SCHRODER P., COHEN M. F., HANRAHAN P.: Wavelet Radiosity. In *Computer Graphics Proceedings, Annual Conference Series, (ACM SIGGRAPH '93)* (1993), pp. 221–230.

[GTGB84] GORAL C. M., TORRANCE K. E., GREENBERG D. P., BATTAILE B.: Modelling the Interaction of Light Between Diffuse Surfaces. *Computer Graphics (ACM SIGGRAPH '84) 18*, 3 (July 1984), 212–222.

[HCA00] HOLZSCHUCH N., CUNY F., ALONSO L.: Wavelet radiosity on arbitrary planar surfaces. In *Rendering Techniques 2000 (11th Eurographics Workshop on Rendering)* (2000), pp. 161–172.

[Hec92] HECKBERT P.: Discontinuity Meshing for Radiosity. In *Third Eurographics Workshop on Rendering* (May 1992), pp. 203–226.

[Hed98] HEDLEY D.: *Discontinuity Meshing for Complex Environments*. PhD thesis, Department of Computer Science, University of Bristol, Bristol, UK, Aug. 1998.

[HS92] HANRAHAN P., SALZMAN D.: A Rapid Hierarchical Radiosity Algorithm for Unoccluded Environments. In *Photorealism in Computer Graphics (Proceedings Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics, 1990)* (1992), pp. 151–171.

[HSA91] HANRAHAN P., SALZMAN D., AUPPERLE L.: A Rapid Hierarchical Radiosity Algorithm. *Computer Graphics (ACM SIGGRAPH '91) 25*, 4 (July 1991), 197–206.

[HWP97] HEDLEY D., WORRALL A., PADDON D.: Selective culling of discontinuity lines. In *Rendering Techniques '97 (8th Eurographics Workshop on Rendering)* (1997), pp. 69–80.

[KH84] KAJIYA J. T., HERZEN B. P. V.: Ray Tracing Volume Densities. *Computer Graphics (ACM SIGGRAPH '84) 18*, 3 (July 1984), 165–174.

[LTG92] LISCHINSKI D., TAMPIERI F., GREENBERG D. P.: Discontinuity Meshing for Accurate Radiosity. *IEEE Computer Graphics and Applications 12*, 6 (November 1992), 25–39.

[LTG93] LISCHINSKI D., TAMPIERI F., GREENBERG D. P.: Combining Hierarchical Radiosity and Discontinuity Meshing. In *Computer Graphics Proceedings, Annual Conference Series, (ACM SIGGRAPH '93)* (1993), pp. 199–208.

[PB95] PATTANAIK S. N., BOUATOUCH K.: Discontinuity Meshing and Hierarchical Multiwavelet Radiosity. In *Graphics Interface '95* (May 1995), pp. 109–115.

[SG94] STEWART A. J., GHALI S.: Fast Computation of Shadow Boundaries Using Spatial Coherence and Backprojection. In *Computer Graphics Proceedings, Annual Conference Series 1994 (ACM SIGGRAPH '94)* (1994), pp. 231–238.

[SGCH93] SCHRODER P., GORTLER S. J., COHEN M. F., HANRAHAN P.: Wavelet Projections for Radiosity. In *4th Eurographics Workshop on Rendering* (June 1993), pp. 105–114.

[SSSS98] STAMMINGER M., SCHIRMACHER H., SLUSALLEK P., SEIDEL H.-P.: Getting rid of links in hierarchical radiosity. *Computer Graphics Forum (Eurographics '98) 17*, 3 (September 1998), C165–C174.

[Stu94] STURZLINGER W.: Adaptive Mesh Refinement with Discontinuities for the Radiosity Method. In *Photorealistic Rendering Techniques (5th Eurographics Workshop on Rendering)* (June 1994), pp. 239–248.

[SWND03] SHREINER D., WOO M., NEIDER J., DAVIS T.: *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.4*. Addison Wesley Professional, 2003, ch. 11. Tessellators and Quadrics.

[WH97] WILLMOTT A., HECKBERT P.: An empirical comparison of progressive and wavelet radiosity. In *Rendering Techniques '97 (8th Eurographics Workshop on Rendering)* (1997), pp. 175–186.