A Discontinuity Meshing Algorithm for Accurate Radiosity[1]

Dani Lischinski *Filippo Tampieri* *Donald P. Greenberg*

Program of Computer Graphics
Cornell University
Ithaca, New York 14853

July 1992

Abstract

We discuss the problem of accurately computing the illumination of a diffuse polyhedral environment due to an area light source. We show how umbra and penumbra boundaries and other illumination details correspond to discontinuities in the radiance function and its derivatives. The shape, location, and order of these discontinuities is determined by the geometry of the light sources and obstacles in the environment. We describe an object-space algorithm that accurately reproduces the radiance across a surface by constructing a *discontinuity mesh* that explicitly represents various discontinuities in the radiance function as boundaries between mesh elements. A piecewise quadratic interpolant is used to approximate the radiance function, preserving the discontinuities associated with the edges in the mesh. This algorithm can be used in the framework of a progressive refinement radiosity system to solve the diffuse global illumination problem. Results produced by the new method are compared with ones obtained using a standard radiosity system.

[1]Appeared in *IEEE Computer Graphics and Applications*, vol. 12, num. 6, November 1992, pp. 25–39.

1 Introduction

Global illumination algorithms can be classified into two main categories: *view-dependent* methods, such as ray-tracing, which compute an image for a particular point of view; and *view-independent* methods, such as radiosity. This work falls in the latter category, and is focused on faithfully simulating the illumination phenomena. More specifically, our goals are to provide accurate and artifact-free solutions without requiring user intervention. This is achieved through automatic meshing, accurate illumination computations, and non-linear interpolation.

Since radiosity techniques were first introduced [1, 2], their applicability has been extended from purely diffuse to partially specular environments [3, 4, 5], to arbitrary bidirectional reflectance functions [6], and their performance has been sped up immensely [7, 8]. Despite this impressive progress, however, radiosity solutions still lack photorealism.

In the real world there is an abundance of subtle and complex illumination details that are hard to simulate, even for the simplest environments. Figure 1 shows a photograph of a cardboard triangle suspended over a matte white floor illuminated by a square diffuse light source. Even in this simple case a number of interesting details are present: there are dark bands extending from the corners of the umbra, and a bright horizontal Mach band along the outer edge of the penumbra near the bottom of the picture.

These features, as well as the boundaries of umbra and penumbra regions, correspond to discontinuities in the *radiance* function and in its derivatives. The shape, location, and order of these discontinuities is determined by the geometry of the light sources and obstacles in the environment, and their correct reproduction is key to achieving higher photorealism.

We describe an algorithm for compactly and accurately capturing the illumination of a diffuse polyhedral environment due to an area light source. The algorithm constructs a *discontinuity mesh* that explicitly represents discontinuities in the radiance function as boundaries between mesh elements. A piecewise quadratic interpolant is then used to approximate the radiance function, preserving the discontinuities associated with the edges in the mesh. This algorithm can be used in the framework of a progressive radiosity system to compute a complete diffuse global illumination solution.

The new algorithm is capable of accurately reproducing fine illumination details (see Figure 2) and it possesses several other important advantages over existing methods: the discontinuity meshing scheme is fully automatic, and it results in solutions which are more view-independent than those produced by current state of the art radiosity systems; it is not limited to constant radiosity patches, so there is no restriction on the size of the patches; the quadratic interpolation scheme results in better approximations using less elements and reduces visual artifacts in the resulting images. Finally, the algorithm is based on the understanding of properties of radiance functions; it is not merely a set of *ad hoc* fixes.

The paper is organized as follows. We start with a review of the common problems with radiosity and a survey of previous attempts to overcome them. In Section 3 we reformulate the global illumination problem as a continuous integral equation rather than as the discrete radiosity equation used in the traditional radiosity method. In Section 4 we discuss the properties of radiance functions, and in Section 5 we derive the new algorithm from these properties and describe it in detail. In Section 6 we present some results produced by the new method and compare them to images produced by a standard radiosity system. Finally, in Section 7 we discuss the pros and cons of our approach and offer directions for future research.

2 Review of Previous Work

It is difficult to accurately simulate global illumination with traditional radiosity systems. Such systems start by meshing the environment into an initial set of elements based on local geometric considerations only; then, as the solution process progresses and more information about the illumination is gained, adaptive subdivision schemes are used to refine regions of high radiosity gradients [9]. The quality of the resulting images is heavily dependent on the size and shape of the initial mesh. This dependence has been recently discussed in the radiosity literature [10, 11]: shadows falling between mesh vertices can be entirely missed; “shadow leaks” and “light leaks” can occur; objects can appear to float in mid-air; shadow areas can be expanded, reduced, or distorted; and shadow boundaries can appear jagged or excessively blurry.

Furthermore, the traditional radiosity formulation assumes a constant radiosity distribution across each patch and element. This assumption is often violated leading to significant inaccuracies [12].

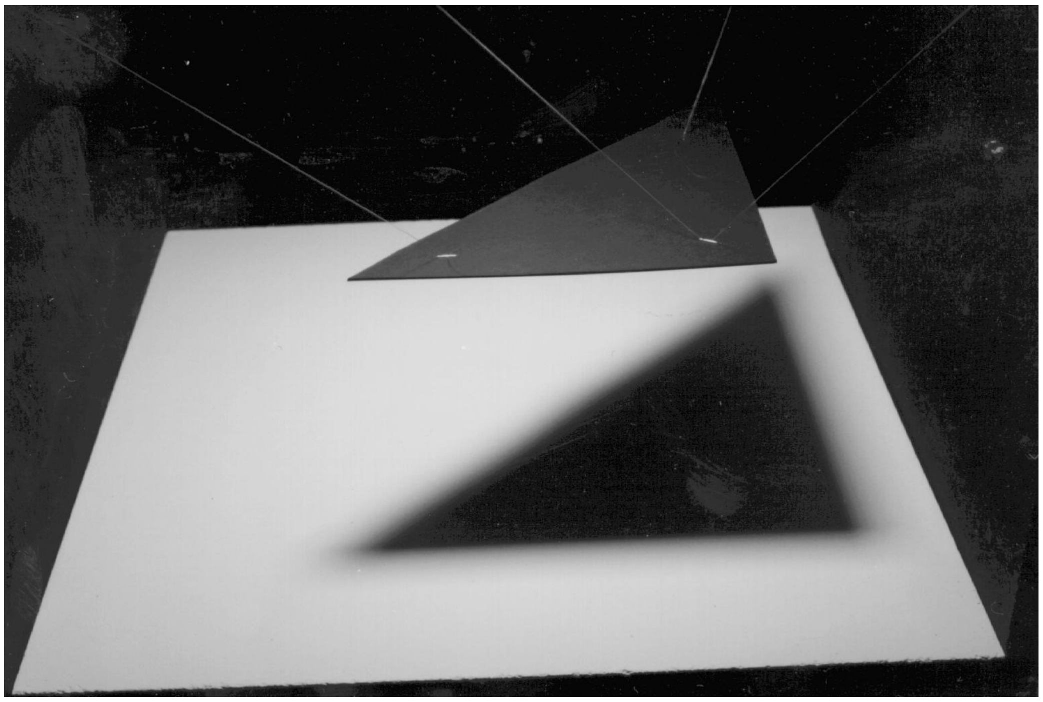


Figure 1: Photograph of a simple environment

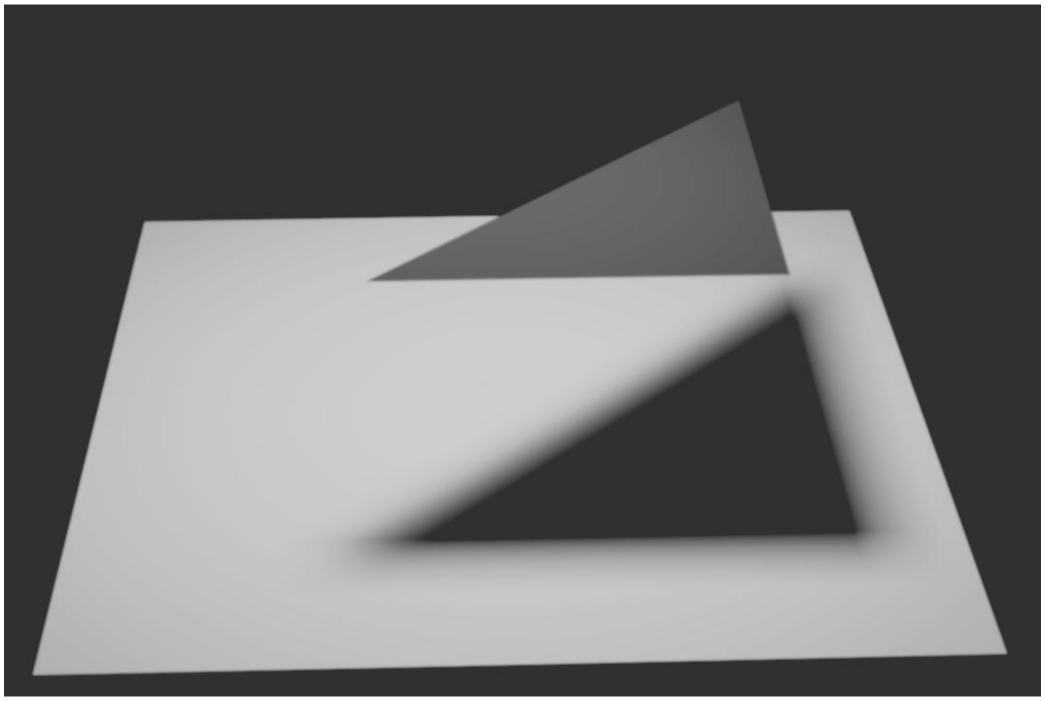


Figure 2: Computer simulation of a simple environment

Lastly, at display, radiosity values are linearly interpolated between the vertices of each mesh element. This often results in undesirable visual artifacts such as Mach banding. As evidenced by Figure 1, some Mach bands are perceived when looking at real environments because of first derivative discontinuities in the actual radiance; the Mach bands introduced by linear interpolation, instead, are due to strong first derivative discontinuities at the element boundaries where the actual radiance function is smooth.

In order to circumvent these problems and produce high-quality images, radiosity systems frequently require intensive user intervention. The density of the mesh elements is often specified on a surface-by-surface basis and repeatedly adjusted through a tedious and time consuming trial-and-error process. Despite these manual adjustments, the resulting radiosity solutions are generally accurate for only a small range of views and thus, are not truly view-independent. For example, zooming in on a shadow is likely to reveal a jagged boundary.

Much of the recent radiosity research is concerned with improving the accuracy of radiosity solutions while reducing the need for user intervention. We briefly survey the results, in order of increasing similarity to our own work.

Hanrahan *et al.* [8] present a hierarchical radiosity algorithm that reduces the number of interactions between elements by an order of magnitude. Elements are automatically and adaptively refined by the algorithm, according to a user-specified error bound. Thus, in addition to drastically reducing computation times, the algorithm also provides automatic meshing.

Baum *et al.* [10] describe a series of clever techniques designed to preprocess an input model into a mesh that meets certain geometrical and topological requirements needed for computing accurate solutions. However, user intervention is not entirely eliminated, since the initial patch size must still be specified.

Both of the above methods use numerical algorithms for occlusion determination and are thus still liable to miss shadows. Also, the mesh boundaries may not be aligned with the shadow boundaries, which is the cause for the jagged appearance of shadows.

Campbell and Fussell [13] approximate each area light source by a set of point light sources. Shadow volumes from each of these points are used to mesh the environment. This method works well for small light sources, but as the number of point sources needed for an adequate approximation grows, subdivision within regions of penumbra becomes too fine. Their approach was later changed to treat area light sources more accurately [14]. Through the use of umbra and penumbra volumes, the environment is classified into unoccluded, umbra, and penumbra regions. Each region is further split into elements, and the illumination at the vertices of each element is computed analytically. To our knowledge, this is the first object-space algorithm to accurately compute shadows cast by area light sources. A weakness of this method seems to be its reliance on numerical optimization for determining the correct element density inside penumbra regions. We suspect that this might become expensive and/or inaccurate for large penumbrae. For instance, the details near the corners of the shadow in Figure 1 might be missed.

The idea of including discontinuities in radiance functions and their derivatives as boundaries in the mesh was developed by Heckbert [15]. Similar ideas were pursued independently by the authors [16]. In both cases, the ideas were implemented and tested in 2D, showing promising results. The current work extends these ideas into a general method in 3D[2].

3 Problem Formulation

Assume that the environment is defined by a collection of surfaces S. Let L_i denote the *radiance function* over surface $s_i \in S$. The function $L_i(x, x'', \lambda)$ gives the light energy flux per unit solid angle per unit projected area leaving surface s_i from point x towards point x'', at wavelength λ. *Radiance* (often named *intensity* in the computer graphics community) is the quantity used for assigning pixel values in image synthesis. Therefore, it is convenient to reformulate Kajiya's rendering equation [18] in terms of radiance [19]:

$$L_i(x,x'',\lambda) = L_i^e(x,x'',\lambda) + \sum_{s_j \in S} \int_{x' \in s_j} \rho_i(x',x,x'',\lambda) L_j(x',x,\lambda) \frac{\cos\theta_i \cos\theta_j}{r^2} v(x,x') dA(x') \tag{1}$$

where

$L_i^e(x,x'',\lambda)$ is the emitted radiance from x towards x'' at wavelength λ;

[2]After this work was completed, it was brought to our attention that Heckbert has also been working on a 3D extension to his work. His results are described in [17].

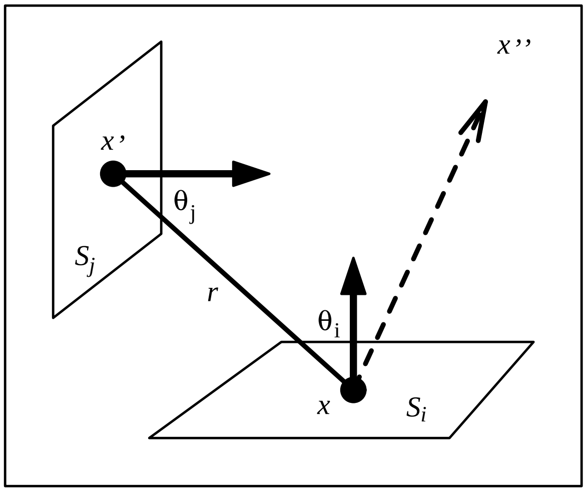


Figure 3: Geometry for Equation (1).

$\rho_i(x', x, x'', \lambda)$ is the bidirectional reflectance distribution function at x, i.e., the ratio of the reflected intensity towards x'' to the energy incident per unit time and per unit area from the direction of x';

$v(x, x')$ is 1 if x and x' are visible to each other, and 0 otherwise;

$dA(x')$ is a differential area element centered at x' (on surface s_j);

r is the distance between x and x';

θ_i and θ_j are the angles between the surface normals at x and x' and the line connecting the two points (see Figure 3).

If all the surfaces are ideal diffuse (Lambertian) reflectors, with constant reflectivity across their areas, then for a single wavelength λ the following simplified equation is obtained:

$$L_i(x) = L_i^e(x) + \rho_i \sum_{s_j \in S} \int_{x' \in s_j} L_j(x') \frac{\cos\theta_i \cos\theta_j}{r^2} v(x, x') dA(x') \quad (2)$$

This equation can also be expressed in terms of radiosity. Recall that the radiosity $B_i(x)$, is the total (hemispherical) light energy flux per unit area leaving surface s_i from point x. It is easy to show by integrating the radiance over the hemisphere that $B_i(x) = \pi L_i(x)$ [2]. The usual discrete radiosity equation can be derived from Equation (2) using the assumption that radiosity is constant over each patch [16].

The method presented in this paper progressively computes a piecewise quadratic approximation to L_i, rather than solving the discrete equation via a piecewise constant approximation, as is done by standard radiosity methods. Hence, radiance is no longer assumed constant across each patch.

4 Radiance Functions

In the previous section the global illumination problem was formulated as one of finding a set of radiance functions (one for each surface in the environment). In this section we will examine various properties of these functions. This will lay out the foundation for the algorithm that we describe in the next section. We limit the analysis to the domain of polygonal environments where the polygons do not interpenetrate.

From Equation (2) we see that the radiance function L_i over surface i is given by a sum of terms, each term representing the radiance contributed to surface i by a different surface j in the environment. To simplify matters we restrict the discussion to the radiance function L_{ij}, i.e., the radiance contributed to surface i by surface j:

$$L_{ij}(x) = \rho_i \int_{x' \in s_j} L_j(x') \frac{\cos\theta_i \cos\theta_j}{r^2} v(x, x') dA(x') \quad (3)$$

Since $v(x,x')$ can only assume values of 0 or 1, we can take it out of the integrand if we limit the range of integration to the parts of s_j visible from x, which we denote by $V(x,s_j)$:

$$L_{ij}(x) = \rho_i \int_{x' \in V(x,s_j)} L_j(x') \frac{\cos\theta_i \cos\theta_j}{r^2} dA(x') \tag{4}$$

Let us assume, for now, that the radiance function across the source is smooth (C^∞), as is the case with primary sources, which typically have constant radiance. We will also assume that x does not lie on the source, i.e., $r \neq 0$. Under these assumptions, the integrand is a smooth function, and hence L_{ij} will be smooth itself, if the visibility of the source s_j is constant or varies smoothly. Abrupt changes in visibility, however, will introduce discontinuities of various orders into L_{ij}. In the rest of this section we will enumerate the possible discontinuities and explain their causes. A similar discussion first appeared in Heckbert's thesis [15], and we therefore adhere to the terminology and notation used there.

We will use the notation that a function L has a D^k discontinuity at x, if it is C^{k-1} but not C^k there. If the radiance over the light source s_j is smooth, the radiance function L_{ij} can have D^0, D^1, and D^2 discontinuities.

D^0 discontinuities

By definition, these are discontinuities in the function itself (value discontinuities). They are introduced by edges or vertices of occluders (or the light source itself) lying on the receiver s_i. Thus, they can occur either along line segments, or in a pointwise fashion. An example is shown in Figure 4.

D^0 discontinuities also occur along shadow boundaries cast by point light sources. In this case no penumbra regions are present, and the transition from the umbra into the unoccluded area is a discontinuous one. For conciseness, we discuss finite area light sources only. The correct treatment of point light sources is simpler and, as will become apparent in what follows, only requires a straightforward extension to our method.

In an environment with m edges, no interpenetrating polygons, and no point light sources, there can be $O(m^2)$ D^0 discontinuities, as it is possible for each of $O(m)$ edges to lie on $O(m)$ faces.

Failure to include D^0 lines into the mesh causes severe light and shadow leaks in radiosity images. These occur since at the rendering stage radiosity values are continuously (linearly) interpolated between points lying on two different sides of the discontinuity [10, 11]. Particular care should be taken at D^0 points, such as points A and B in Figure 4. The radiance function at these points is singular, and the points cannot be assigned a single radiosity value. To our knowledge, no method to date treats D^0 points correctly. We offer a solution to this problem in Section 5.3.

D^1 and D^2 discontinuities

These are discontinuities in the first and second derivatives of the function, respectively. They are caused by edges and vertices of occluders that do not touch the receiver, but intervene between it and the light source, causing abrupt changes in the visibility of the source. These changes in visibility (also known as *visual events*) have been studied in the computational geometry and computer vision literature [20, 21]. The visual events in a polyhedral environment can be classified into two types: edge-vertex (EV) events and edge-edge-edge (EEE) events [20].

EV events occur on a subset of the plane defined by an edge and a vertex fully or partially visible to each other. The set of points on that plane from which the vertex can be seen coinciding with the edge is called the *critical surface*. Visibility changes on a receiver occur along the curves of intersection between the receiving plane and the critical surfaces. These *critical curves* are line segments in the case of an EV event. They correspond to either D^1 or D^2 discontinuities, as demonstrated in Figures 5 and 6. D^1 discontinuities are perceived as Mach bands on the receiver. Strong D^2 discontinuities are noticeable as well.

EEE events occur on a subset of a ruled quadric surface defined by the family of lines that go through three skew edges each visible to each other. The critical surface is the set of all the points from which the three participating edges can be seen intersecting in one point. The critical curves are conics and generally correspond to D^2 discontinuities as demonstrated in Figure 7.

In the examples shown in the figures all the visual events involve either a vertex or an edge of the source. However, visual events can be defined by edges and vertices belonging to any object in the environment. Thus, in an environment with m edges, there can be $O(m^2)$ EV critical surfaces, and $O(m^3)$ EEE critical surfaces. In general, the event will cause a discontinuity in L_{ij} if it is defined by edges and vertices located between the receiver i and the source j and if it is visible from both.

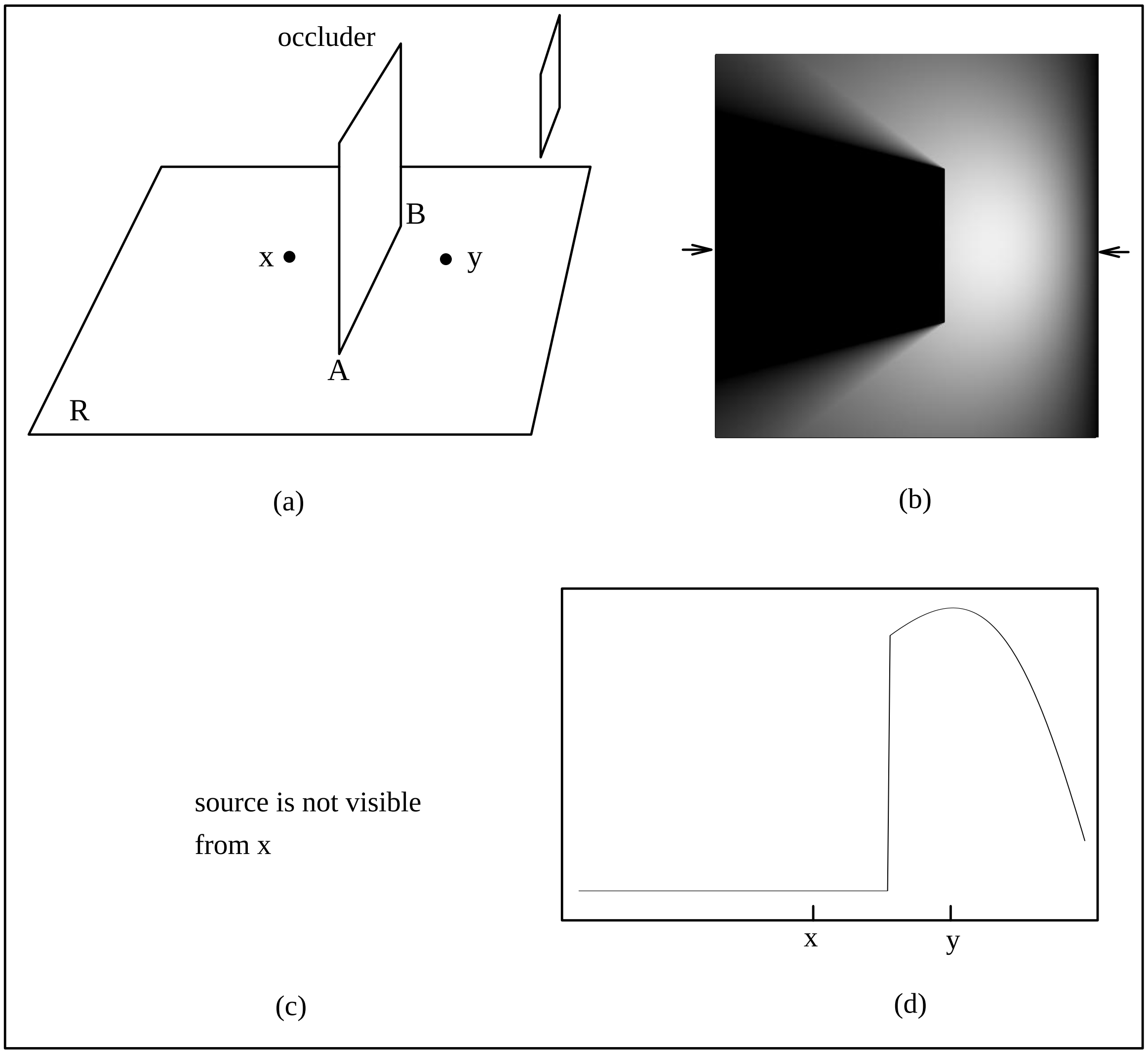


Figure 4: An example of a D^0 discontinuity. (a) Edge *AB* of the occluder lies on the receiver *R*. (b) The radiance function over *R*. (d) The radiance function along the line through *x* and *y*.
Points on *R* immediately to the left of *AB* cannot see the source and the radiance there is zero. However, immediately to the right of *AB*, the entire source is visible, and the radiance there is nonzero. Thus, the radiance function is discontinuous along *AB*. The points *A* and *B* are points of singularity in the radiance function.

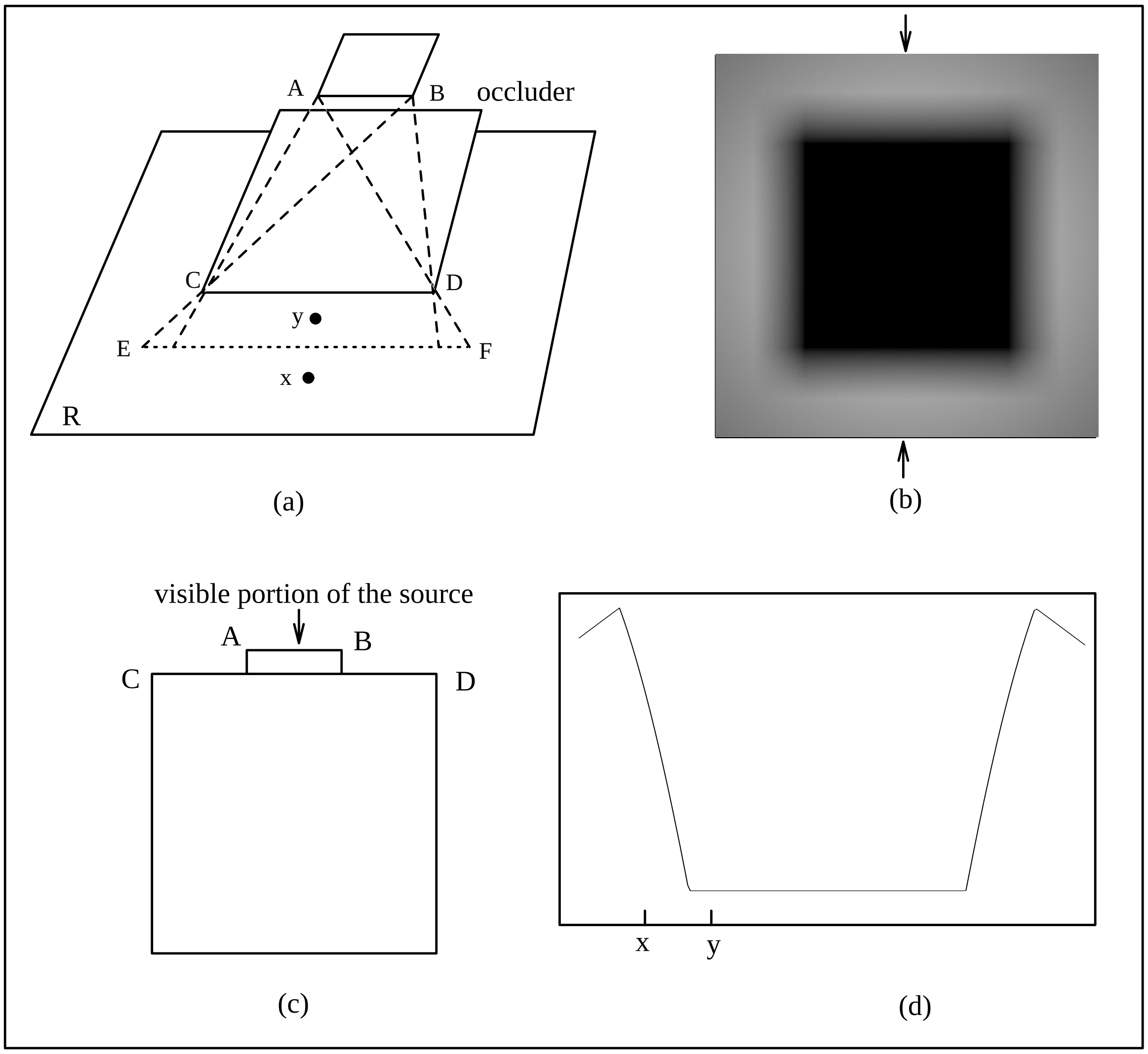


Figure 5: A D^1 discontinuity caused by edge-vertex (EV) events. (a) Edge AB of the light source is coplanar to edge CD of the occluding polygon. The plane in which the two edges lie intersects the receiving plane along EF. (b) The radiance function over R. (c) The occluder and the light source as seen from x. (d) The radiance function along the line through x and y.
From point y on R, none of the source is visible, hence the radiance there is zero. As we move from y towards x, part of the source adjacent to AB becomes revealed. The visible area grows linearly in the displacement fromEF towards x. Thus, along EF the radiance function has a D^1 discontinuity. In this example there are in fact two partially overlapping VE events, one involving vertex A and the other involving vertex B.

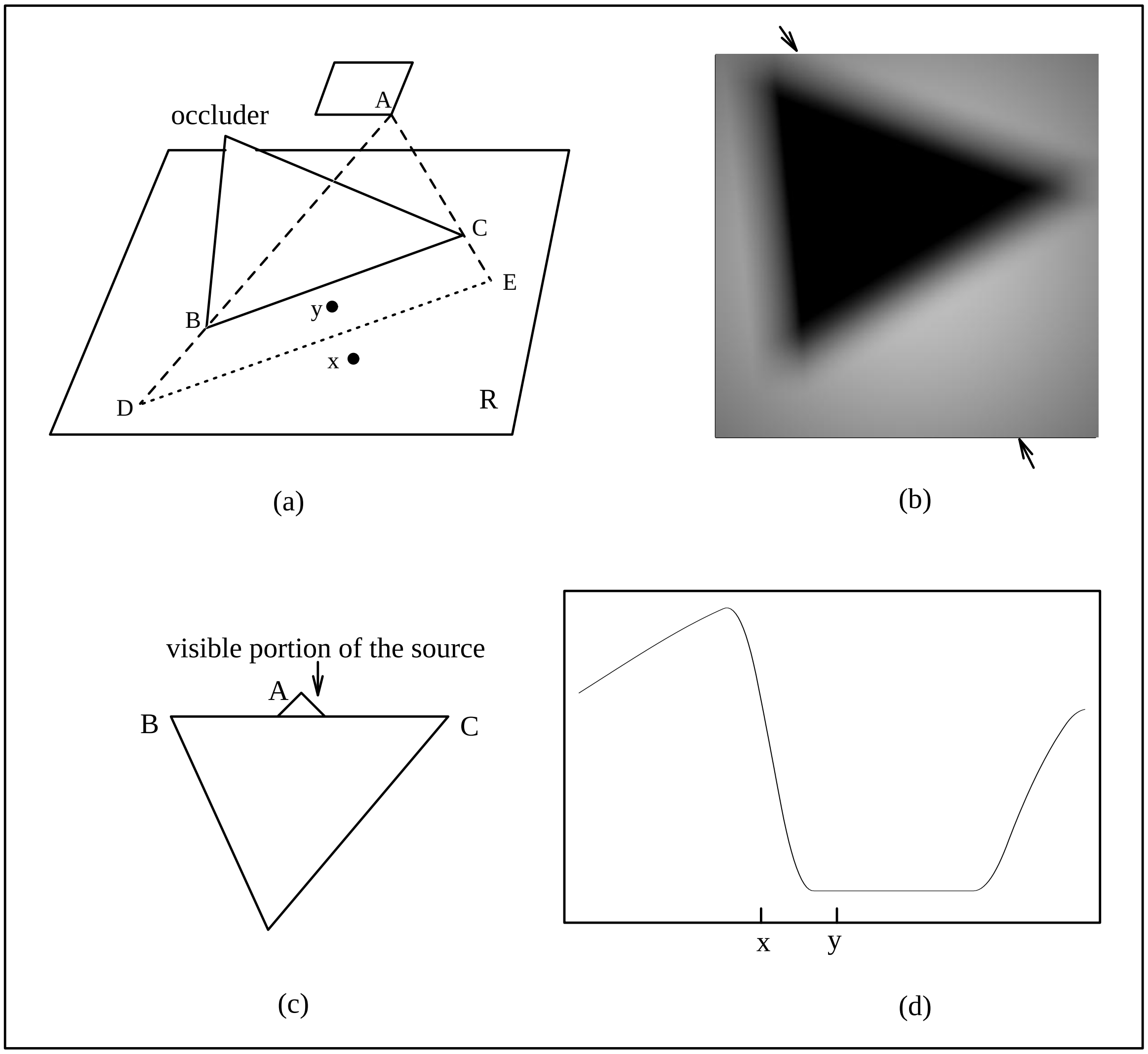


Figure 6: A D^2 discontinuity caused by an edge-vertex (EV) event. (a) The critical surface defined by vertex A and edge BC intersects the receiving plane along DE. (b) The radiance function over R. (c) The occluder and the light source as seen from x. (d) The radiance function along the line through x and y.
From point y on R none of the source is visible, hence the radiance there is zero. As we move from y towards x, part of the source adjacent to vertex A becomes revealed. The visible area grows quadratically in the displacement from DE towards x. Thus, along DE the radiance function has a D^2 discontinuity.

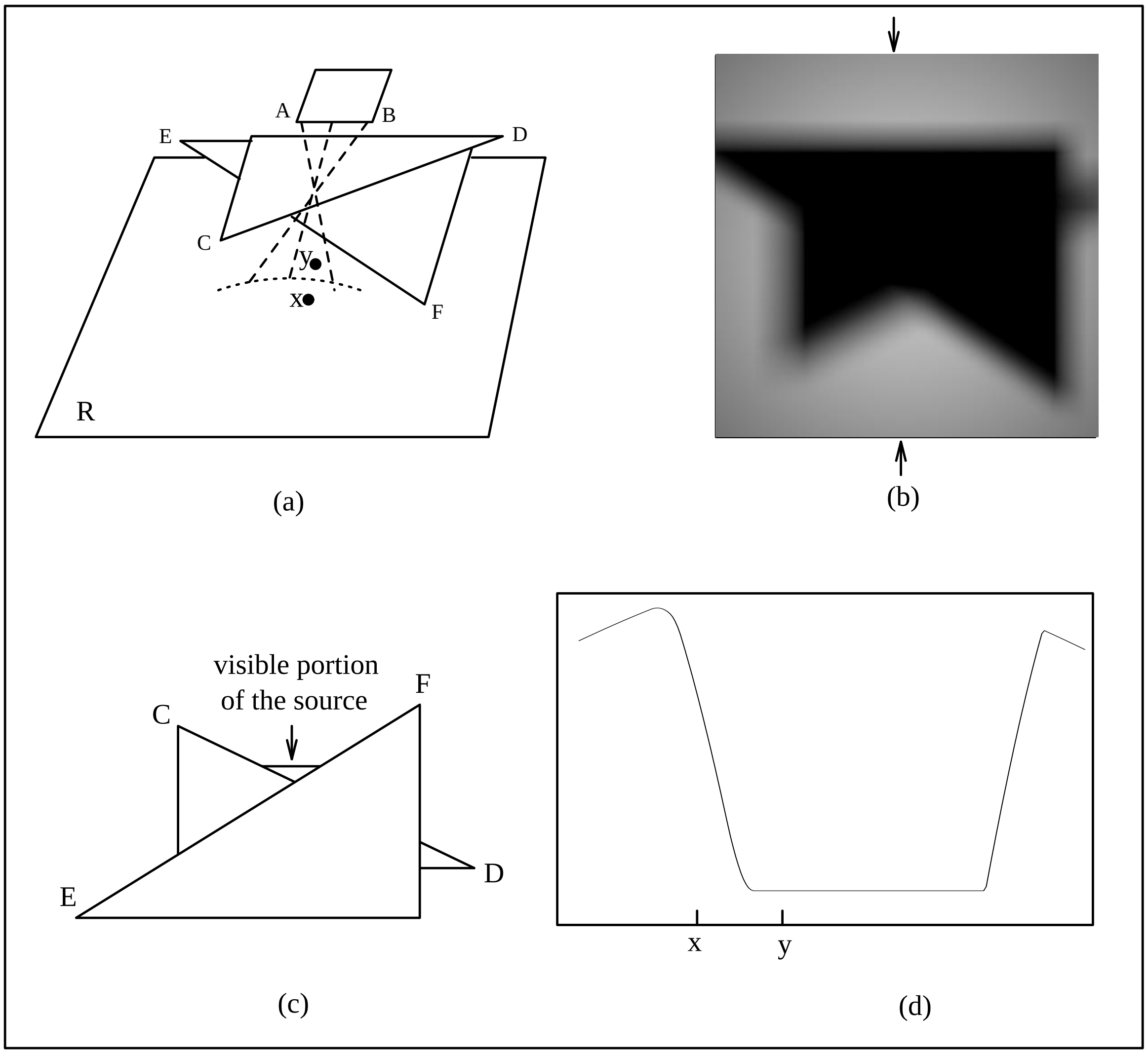


Figure 7: An edge-edge-edge (EEE) event. (a) The quadric critical surface defined by the three edges *AB*, *CD*, and *EF* intersects the receiver, resulting in a conic critical curve (shown dotted). (b) The radiance function over *R*. (c) The occluder and the light source as seen from *x*. (d) The radiance function along the line through *x* and *y*.
As we move from *y* towards *x*, part of the source becomes revealed. A displacement from the critical curve towards *x* results in quadratic growth in the visible source area, hence the discontinuity along that curve is D^2. This example illustrates that when several occluding obstacles are involved, the boundaries between umbra and penumbra regions on a receiver may be curved.

Propagation of Discontinuities

In the preceding discussion we have assumed that the radiance over our source is smooth. However, the radiance over secondary sources can have discontinuities of various orders, as we have just seen. The effect of these discontinuities was summarized by Heckbert [15] as the “Discontinuity Propagation Law”, which states that D^k discontinuities on the source can result in D^{k+1} and D^{k+2} discontinuities on the receivers. In general, discontinuities of higher orders are less noticeable than low order ones.

5 The Discontinuity Meshing Algorithm

In the previous section we have shown that the most significant changes in the radiance function across a receiving surface occur due to discontinuities along critical curves. Traditional radiosity meshes ignore these curves; thus, when a mesh element is crossed by a critical curve, the linear interpolation fails to accurately reproduce the radiance function, resulting in poorly defined shadow boundaries and other artifacts already mentioned earlier. We will use the ability to predict the location and the order of discontinuities to design a new algorithm less susceptible to the problems of traditional radiosity.

Let us start by considering the problem of accurately computing the radiance function L_{ij} contributed from source j to receiver i. To capture the behavior of this function we compute the critical curves caused by visual events involving the source j. We then construct a *discontinuity mesh* over surface i that resolves all these critical curves: each critical curve is explicitly represented in the mesh by means of one or more edges. Each edge is labeled with the appropriate degree of discontinuity. Since the discontinuities occur on mesh edges, the radiance function inside each element is well behaved and can in most cases be captured by a relatively small number of samples. Then the contributed radiance is computed at a set of selected locations within each mesh element, and elements are adaptively subdivided where necessary to yield a more accurate approximation. An approximation to L_{ij} is reconstructed by using quadratic interpolation over each element. The interpolation scheme preserves the appropriate degree of discontinuity across each mesh edge.

In order to make use of this algorithm in environments containing multiple light sources, we need a scheme for combining contributions due to different sources. Traditional radiosity systems use a single mesh throughout the entire simulation. The mesh can be adaptively subdivided in an attempt to capture fine illumination details, but basically the same set of sample locations is used to compute the contribution from all the light sources, both primary and secondary.

Such a scheme would be impractical for discontinuity meshing, since every surface in the environment may become a source, participating in a different set of visual events. Accounting for all these visual events would result in a very large number of critical curves and very complex meshes. Computing the radiance contribution from every source to such a mesh would be very costly.

Our solution, instead, consists of building a separate discontinuity mesh for each source j using only the critical curves resulting from visual events involving this source. In this way, the number of points at which L_{ij} is computed is substantially reduced. The result is a modified progressive refinement radiosity algorithm [7]. At each iteration we select a new source j, build the corresponding discontinuity mesh and use it to compute the contribution of this source. This contribution is then added to a cumulative radiance function and the discontinuity mesh is disposed of. Similarly to the single contribution, the cumulative radiance function is also stored as a discontinuity mesh. As will be shown in the results section, the accumulation process takes up only a small percentage of the total computation time and thus does not compromise the advantages gained by using a separate discontinuity mesh for each source.

A pseudocode outline of the modified progressive radiosity algorithm is given in Figure 8. The key steps of the algorithm are depicted by way of a simple example in Figure 9.

In the following subsections, we describe the major steps of the new method in greater detail and present the main data structures and algorithms used in our particular implementation.

5.1 Locating the Discontinuities

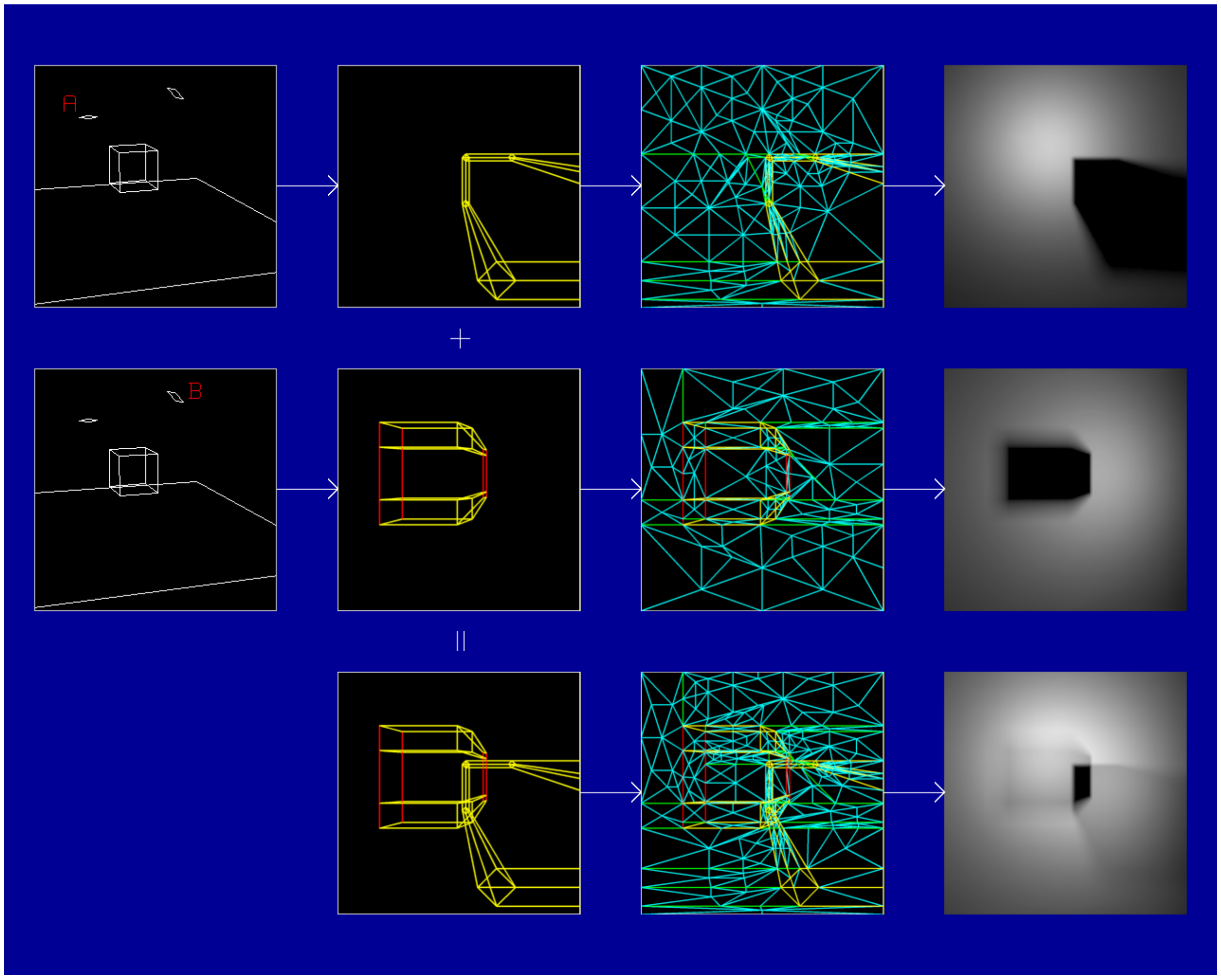
Recall that D^0 discontinuities are generated by contact between surfaces and are therefore independent of the shooting source. These discontinuities are precomputed before the iterative solution begins.

The D^1 and D^2 discontinuities are generated by EV and EEE events. In Section 4 it was shown that there can be $O(m^2)$ EV events and $O(m^3)$ EEE events in an environment with m edges. Thus, an attempt to resolve all the

```
Initialize:
     FOR all surfaces $S_i$ {
          /* Set total and unshot radiance to the emitted radiance */
          $L_i$ := $L_i^e$;
          $\Delta L_i$ := $L_i^e$;
     }
DO until convergence {
     Select shooting surface $S_j$ (with maximum unshot power)
     Shoot($S_j$) /* (compute $L_{ij}$ for all $i$) */
     FOR each receiving surface $S_i$ {
          /* Update the total and unshot radiance functions */
          $L_i$ := $L_i + L_{ij}$;
          $\Delta L_i$ := $\Delta L_i + L_{ij}$;
     }
     /* Set the unshot radiance of $S_j$ to 0 */
     $\Delta L_j$ := 0;
}
```

```
PROCEDURE Shoot($S_j$):
     Find lines of discontinuity due to $S_j$
     Construct a discontinuity mesh on every $S_i$
     FOR each receiving surface $S_i$ {
          FOR each receiving element $E_k$ of $S_i$ {
               Compute the $L_{ij}$ at sample locations within $E_k$
               Adaptively subdivide $E_k$, if needed.
          }
     }
```

Figure 8: The modified progressive radiosity algorithm



(a) (b) (c) (d)

Figure 9: The diagram illustrates how discontinuity meshing is used to compute the combined illumination from two light sources in the presence of a simple obstacle (the cube floating above the floor). Columns (b), (c), and (d) show a plan view of the floor. Row 1. (a) Surface A is selected as the current source. (b) Discontinuity boundaries due to source A are computed. D^1 and D^2 discontinuities are drawn respectively in red and yellow. (c) A triangular mesh which resolves the discontinuity boundaries is produced. (d) The source contribution $L_{\text{floor},A}(x)$ is computed at sample locations within the mesh to yield the interpolant $\tilde{L}_{\text{floor},A}$. Row 2. The process is repeated anew with surface B selected as the source. Row 3. (b) The discontinuity boundaries due to A and B are combined. (c) A new triangular mesh with the combined discontinuity boundaries is produced. (d) Interpolants $\tilde{L}_{\text{floor},A}$ and $\tilde{L}_{\text{floor},B}$ are evaluated at sample locations within the new mesh and the sample values are added to yield the combined contribution of A and B.

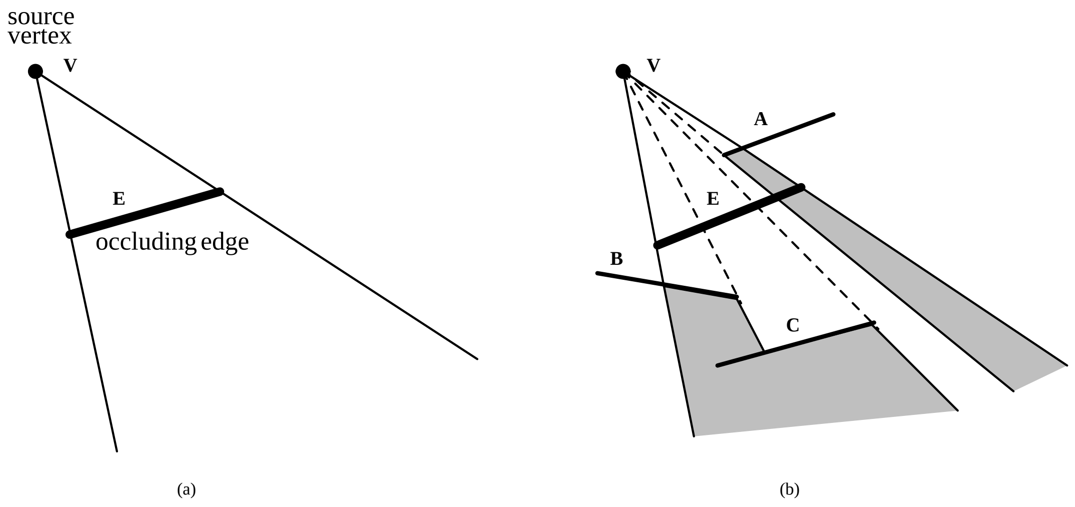


Figure 10: (a) A wedge corresponding to a VE event. (b) Interaction of the wedge with the surfaces in the environment. The wedge is clipped against surfaces A, B, and C, but only surfaces B and C gain a new discontinuity boundary. Discontinuities are not added to surfaces intersecting the wedge in the region enclosed between V and E, since the visual event is not visible from such surfaces. The dark areas show the clipped portion of the wedge.

corresponding discontinuities would result in an algorithm of overwhelming cost. Therefore, we have decided to limit the computation to only EV visual events in which either the vertex or the edge belong to the light source. The number of these events is only $O(m)$ and their corresponding critical curves are line segments. In particular, these segments include all the boundaries separating totally unoccluded regions from partially occluded ones, as well as additional important discontinuities inside the penumbrae. The number of visual events can be further reduced by considering only silhouette edges and vertices.

Determining the order of discontinuity corresponding to an event is simple. Let v and e be the defining vertex and edge, respectively. If v is the endpoint of an edge e' that is coplanar with e, the event should be classified as D^1; otherwise, it should be classified as D^2.

We distinguish between the EV events involving a source vertex and the ones involving a source edge. The former kind will be henceforth referred to as VE events. A VE event is represented by a planar wedge of infinite extent with its apex at the originating vertex and bounded by rays through the endpoints of the occluding edge (see Figure 10(a)). The discontinuities caused by a VE event are found by intersecting its wedge with the surfaces that are visible from the location of the apex. If a surface faces the apex of the wedge, then the segment of intersection is inserted in the surface mesh as a discontinuity edge. If a surface is closer to the apex of the wedge than the occluding edge, then no edge should be added to the surface mesh (see Figure 10(b)).

In our implementation, the environment is stored in a BSP-tree [22]. In order to process a VE event, the surfaces in the BSP-tree are visited in front-to-back order along the wedge, starting from its apex. If a surface is intersected, the wedge is clipped against it and only the unclipped parts of the wedge are traced further. The traversal can stop as soon as all portions of the wedge have been clipped.

An EV event is represented by a slightly more complex wedge (see Figure 11(a)), but intersecting it with the environment requires only minor changes to the algorithm used to process VE events (see Figure 11(b)).

Note that additional events may result from discontinuities on the light source. From the Discontinuity Propagation Law, it follows that, in order to capture discontinuities of order less than or equal to two, we must consider events involving D^0 and D^1 discontinuities on the source as well as those involving the source edges and vertices.

5.2 Constructing a Discontinuity Mesh

The discontinuity mesh on each receiver is represented by a data structure called a *Discontinuity Mesh Tree*, or *DM-tree* for short. A DM-tree consists of two parts: a two-dimensional BSP-tree and a data structure of topologically interconnected faces (elements), edges, and vertices. Each inner node of the tree contains the line equation of a discontinuity segment. Each leaf contains a pointer to a face bounded by the intersection of halfspaces encountered

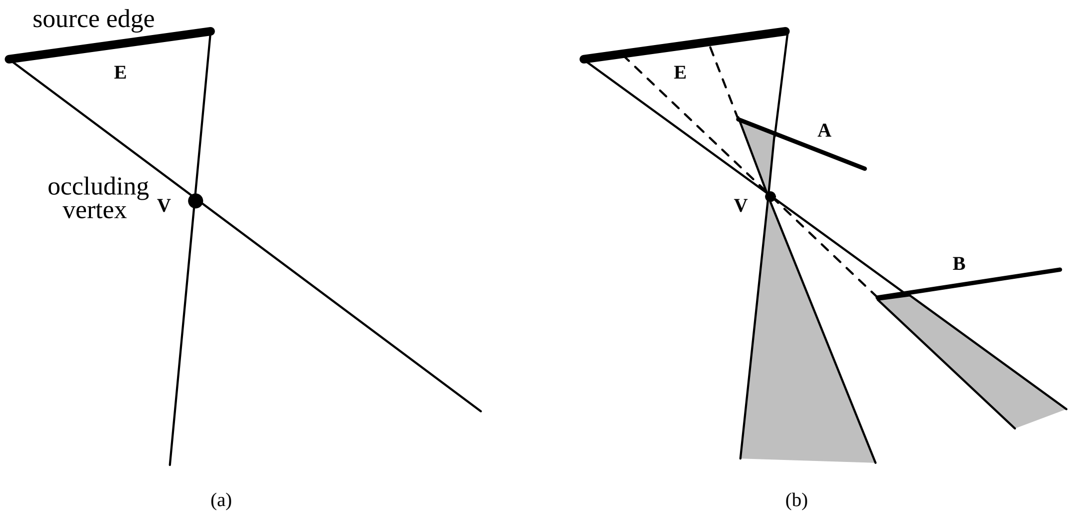


Figure 11: (a) A wedge corresponding to an EV event. (b) Interaction of the wedge with the surfaces in the environment. The wedge is clipped against surfaces A and B, but only surface B gains a new discontinuity boundary. Discontinuity boundaries are not added to surfaces intersecting the wedge in the region enclosed between E and V, since the visual event is not visible from such surfaces. The dark areas show the clipped portion of the wedge.

along the path down from the root of the tree. This data structure is constructed incrementally: each discontinuity segment is filtered down the tree, eventually splitting only the intersected leaves. The faces attached to these leaves are also split, and the newly added edges are labeled with the appropriate order of discontinuity.

The topological mesh is arranged similarly to a winged-edge data structure so that no duplication of edges and vertices is necessary and each element can easily determine its neighbors. In this way the contribution of the source can be computed once on each vertex and shared between the incident elements. Since edges are shared between neighboring elements, no T-vertices are introduced when elements are split.

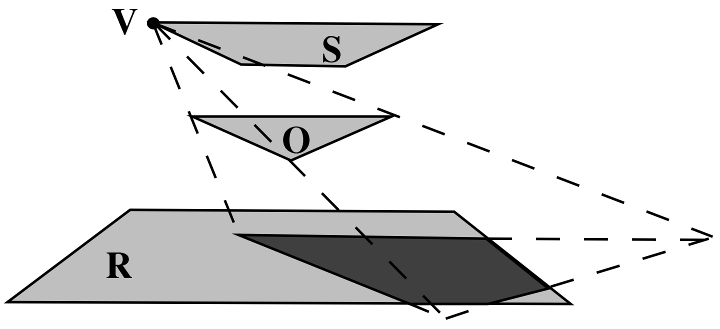
Two-dimensional BSP-trees are also used to represent radiosity meshes by Campbell and Fussell [13]. Note however, that the elements stored at the leaves of their *element BSP-tree* are not topologically connected. This results in redundant illumination computations, as well as in a need for a separate pass to eliminate T-vertices.

Figure 12 shows an example of how a DM-tree is constructed incrementally from a sequence of visual events. The simple configuration given in (a) shows the discontinuity lines on a receiving surface **R**, generated by the interaction of vertex **V** of source **S** with obstacle **O**. Initially, (b) the DM-tree of the receiver consists of a single node pointing to a single face F. When the first edge of the obstacle is projected on the receiver, (c) the line equation a of the resulting discontinuity line is used to split face F into faces F_1 and F_2, and the DM-tree is updated so that its root now stores line equation a and points to the two new faces on either side of a. Notice that two edges have been created along line a: the bottom one corresponds to the actual discontinuity segment, while the top one is due to our splitting scheme and is not marked as a discontinuity. Next, the second VE event is processed (d). A new discontinuity segment is generated and the DM-tree is searched from the root to determine whether any face is crossed by this segment. Face F_2 is found and is therefore split into faces F_{21} and F_{22}. Finally, the third VE event is processed to yield the DM-tree depicted in (e).

5.3 Sampling and Reconstruction on a Receiver

Once the discontinuity mesh is built, sample locations must be chosen at which to compute the radiance contribution of the current light source. The contributed radiance function is then reconstructed from the samples.

The initial faces created while constructing the discontinuity mesh may have many vertices. Reconstructing a smooth radiance function across such general elements while meeting the boundary conditions is difficult. Therefore, each region is triangulated first. To triangulate an element, we look for the longest diagonal and then connect its midpoint to all the vertices of such an element. Since our DM-tree partitions the receiver into convex regions, this simple scheme is guaranteed to yield a valid triangulation.



(a)

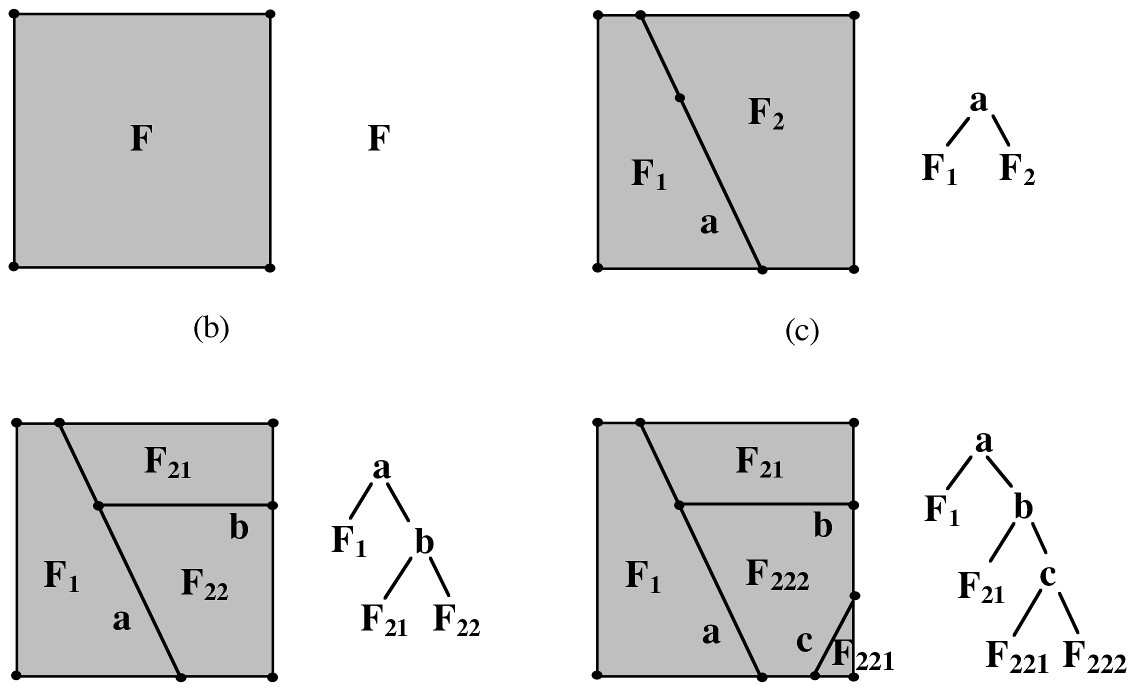


Figure 12: Incremental construction of a DM-tree

Triangular elements of different orders can now be used to represent the radiance function. Constant and linear elements are only able to resolve D^0 and D^1 discontinuities respectively and generate annoying visual artifacts such as blockiness and Mach bands. Higher order elements can correctly resolve D^0, D^1, and D^2 boundary discontinuities, and can more accurately approximate the radiance function inside each element.

We use quadratic elements with each triangle defined by six radiance values: one at each vertex and one at the midpoint of each edge. Rather than actually computing six values for each triangular element, radiance values are shared between neighboring elements across non-D^0 edges. Correct treatment of D^0 edges and vertices requires some attention. The radiance function is not uniquely defined along these edges and vertices and the radiance values in their vicinity depend on the side from which the edge is approached. The correct values for a given element are the limit values of the radiance function at the edge midpoints and vertices when moving towards them from inside the element. In practice, a good approximation to these limit values can be computed by slightly displacing the sample locations corresponding to the vertices and the midpoint of the edge inward, so that they lie strictly inside the element.

A special problem is posed by elements which are incident on a D^0 vertex V, but do not contain D^0 edges. There is a need to correctly represent the singularity that the radiance function may have at V. Assigning a single radiance value to V may cause D^0 discontinuities in the interpolated function along the edges incident on it. To solve this problem we fit a degenerate bi-quadratic patch over the triangular domain of the element. The degenerate patch has three control points corresponding to V and thus a range of values can be represented simultaneously there, while maintaining continuity along the incident edges.

Our implementation correctly resolves D^0 and D^1 discontinuities. The actual radiance function is C^1 continuous across D^2 boundaries. Currently, we do not impose C^1 continuity across these boundaries. It is possible to do so using the piecewise cubic interpolation scheme described by Salesin *et al* [23].

5.4 Computing the Source Contribution to a Point

The radiance contributed by a source s_j to a point x was given by Equation (4). We distinguish between the case where s_j is completely visible and the case where s_j is partially occluded from x.

The Unoccluded Case

If the unshot radiance across the source is constant (as may be the case with primary light sources), the computation is the same as in any other radiosity method. The form-factor from the source to a differential receiving area is computed and is multiplied by the constant radiance value. A number of methods exist for computing form-factors [24, 1, 25]. In the case of a polygonal source, the form-factor can be computed analytically [24, 26]. Because of its high accuracy, this was the method we chose to use in our implementation.

In most cases, however, the radiance across the shooting source is not constant. We use an algorithm for accurately computing the contribution from non-constant sources similar to the one described in [12]. For each point, the algorithm adaptively splits the area of the source, until the errors caused by the constant radiosity assumption over each piece are sufficiently small. The radiant power for each piece is estimated by numerically integrating the radiance function over its area. The contribution of each piece is the product of the radiant power and the corresponding form-factor. These contributions are added together to yield the total source to point contribution.

The Partially Occluded Case

If the source is partially occluded from the receiving point, then in order to obtain accurate results the visible parts of the source are first computed. Polygons which intersect the frustum defined by the receiving point and the source are projected onto the supporting plane of the source, clipping away the occluded portions. Then, the same algorithm used in the unoccluded case is applied to each of the source's visible parts. A similar hidden surface algorithm was first used by Nishita and Nakamae [27] for analytically calculating the illumination from a source of constant emission. Point sampling algorithms for testing the visibility of the light source [8, 25] can also be used, but they are not nearly as accurate, and are prone to aliasing.

5.5 Adaptive Subdivision

The radiance inside each initial triangle is not guaranteed to be accurately captured by a simple quadratic function. When necessary, the accuracy of the approximation can be improved by adaptively subdividing the elements of the mesh. In order to design a refinement algorithm, one has to decide when and how a mesh element should be subdivided.

Intuitively, a mesh element should be subdivided whenever its approximation to the correct radiance function falls below the desired accuracy according to a predetermined error metric. Our implementation uses the infinity-norm error metric, which is defined as the maximum absolute error over the element. We approximate the error by computing the radiance at the centroid of the element and taking the difference between this value and that provided by the interpolant for the same location. This error estimate is not always reliable and can cause the adaptive subdivision to stop prematurely. In practice, however, this strategy has yielded good results.

For mesh subdivision, we have adopted the 2-triangles mesh refinement algorithm presented by Rivara [28]. This refinement strategy has a number of desirable properties, the most interesting being that the refined mesh tends to be made of well shaped triangles. More precisely, it yields triangles whose longest edge is shorter than twice the length of either of the other edges [28]. This property is important because well shaped elements can reduce errors in the approximation [10].

5.6 Combining Radiance Functions

The previous subsections have shown how to construct a piecewise quadratic interpolant $\tilde{L}_{ij}$ approximating the contribution L_{ij} from a source j to a surface i. Here, we show how to combine the contributions of different light sources to construct an interpolant $\tilde{L}_i$ approximating the total radiance function L_i on surface i.

Our progressive radiosity system refines $\tilde{L}_i$ incrementally by adding in at each iteration the latest contribution $\tilde{L}_{ij}$:

$$\tilde{L}_i^{\text{new}} := \tilde{L}_i^{\text{old}} + \tilde{L}_{ij}$$

In this way, only the storage used to represent the current $\tilde{L}_i$ needs to be maintained, while that required by the $\tilde{L}_{ij}$ is released at the end of each iteration.

Since for each source j the discontinuity mesh of the interpolant $\tilde{L}_{ij}$ is typically substantially different from the meshes used for previous sources, the problem of merging contributions is not trivial.

In order to compute $\tilde{L}_i^{\text{new}}$ we start by creating a new discontinuity mesh containing all the significant discontinuity boundaries from $\tilde{L}_i^{\text{old}}$ and $\tilde{L}_{ij}$. These boundaries can be identified by inspecting the corresponding discontinuity meshes. The new mesh is then triangulated and quadratic elements are computed as described in subsections 5.3 and 5.5. Note however, that the value at any point p is given by $\tilde{L}_i^{\text{old}}(p) + \tilde{L}_{ij}(p)$. Thus, rather than integrating over the source at each sample location of the new mesh, we only need to evaluate $\tilde{L}_i^{\text{old}}(p)$ and $\tilde{L}_{ij}(p)$. Each value is computed by locating the element containing p in the appropriate mesh and evaluating the quadratic interpolant.

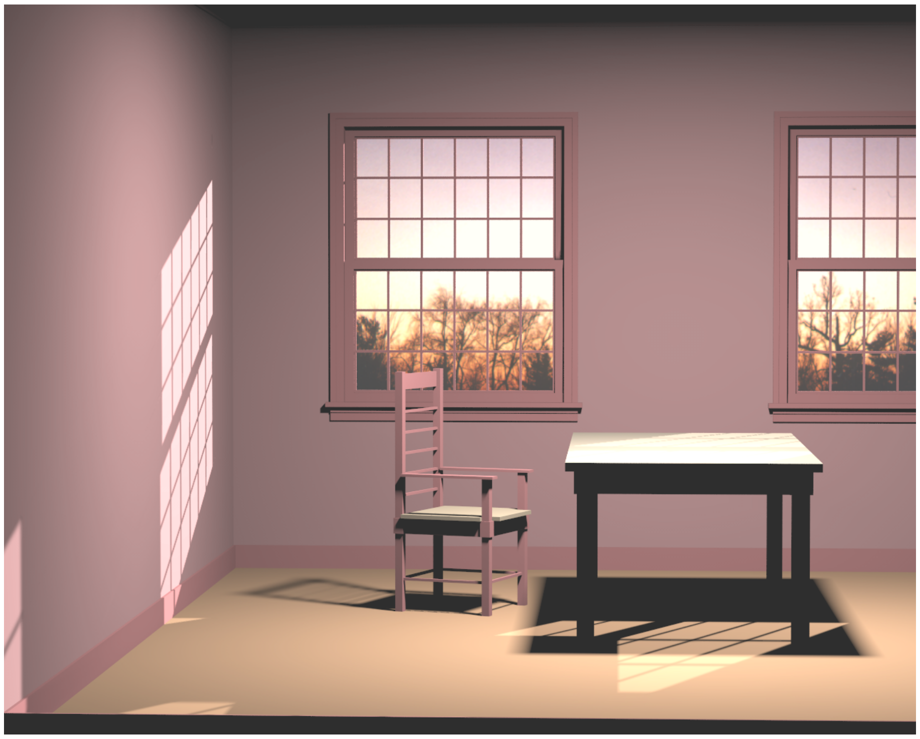
Since there is no need to process visual events nor to compute the contribution of a source at each location, constructing the new interpolant $\tilde{L}_i^{\text{new}}$ is typically much faster than computing the contribution $\tilde{L}_{ij}$.

Our scheme for combining radiance functions is essentially a resampling method, and therefore a certain loss of accuracy can be expected. Notice, though, that since the important critical curves are reproduced in the new discontinuity mesh, the resampling occurs inside regions where the radiance function can be considered smooth, and thus the added error is small. Furthermore, since the time spent combining radiance functions constitutes a very small part of the total simulation time, we can afford to increase adaptive subdivision so as to provide the desired level of accuracy.

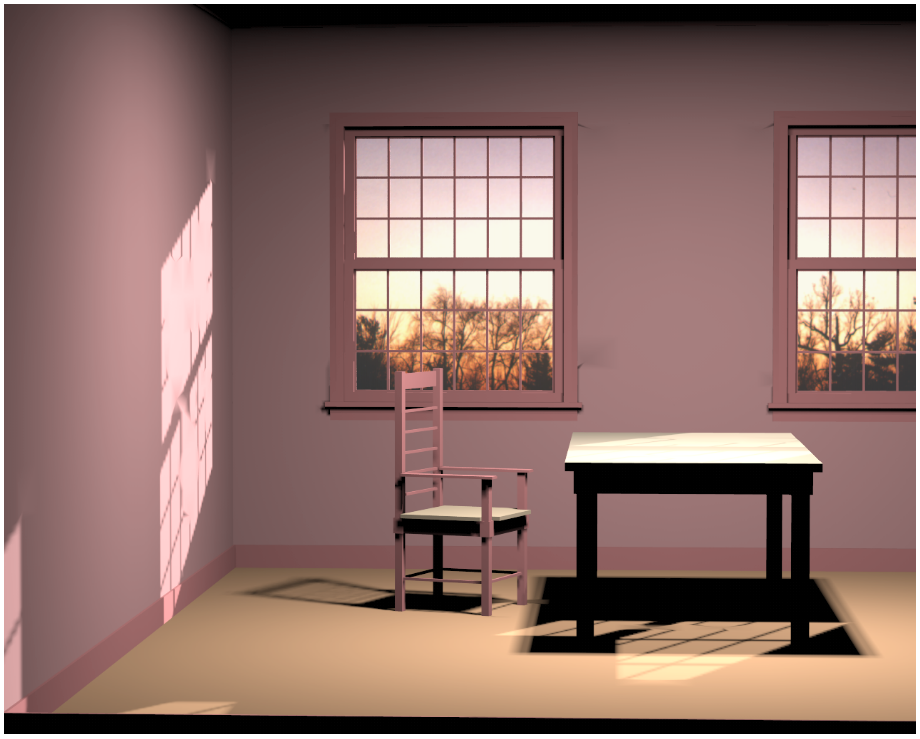
6 Results

In this section, a sample solution obtained by discontinuity meshing radiosity (DMR) is compared with one obtained by a standard radiosity system (SR).

The standard radiosity system we have used is a state of the art production system that was implemented at the Program of Computer Graphics [29]. This system uses ray-tracing to compute form-factors [25]. Adaptive subdivision is used to capture shadows. Users have control over parameters such as initial patch size, minimum element size and the tolerance used for adaptive subdivision. Meshing can be controlled both globally and on a surface-by-surface basis.

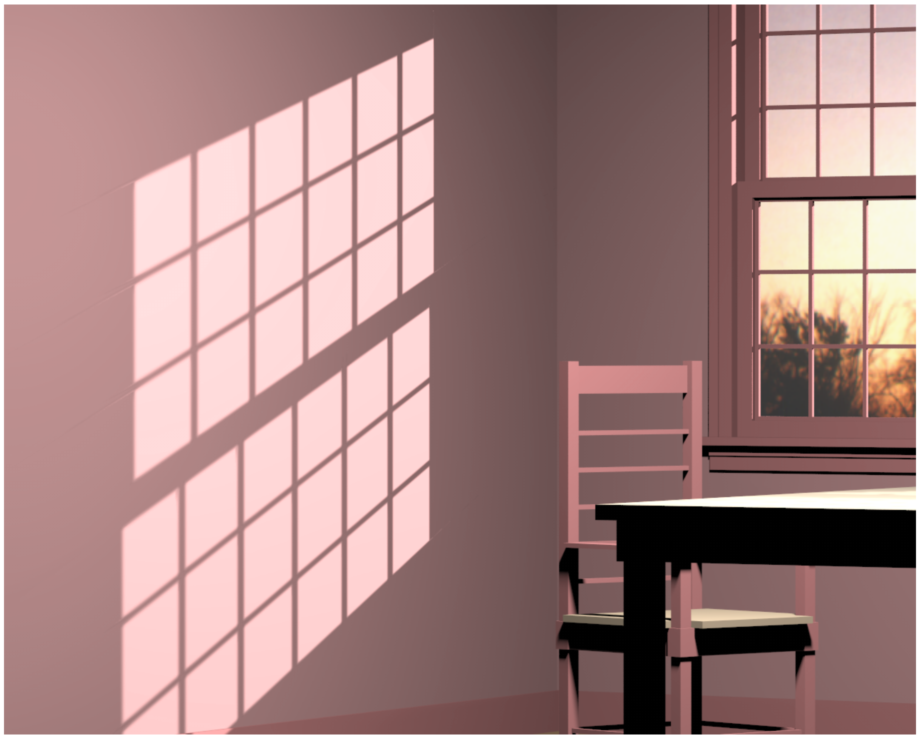


(a)



(b)

Figure 13: A room with a view. Image (a) is rendered using a solution produced by discontinuity meshing radiosity. Image (b) is rendered from a standard radiosity solution.


(a)

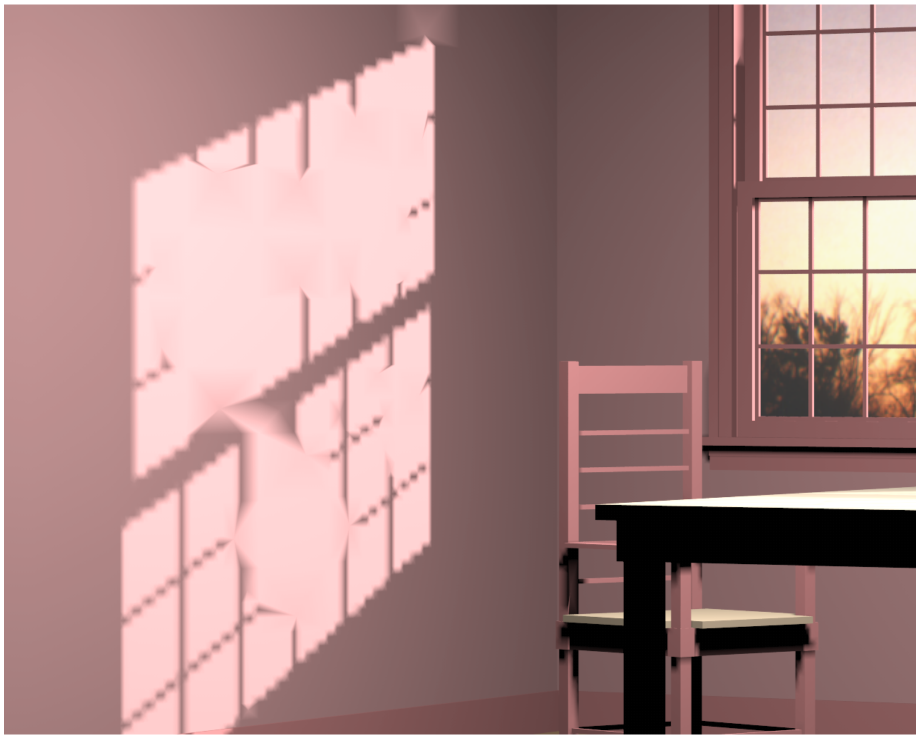
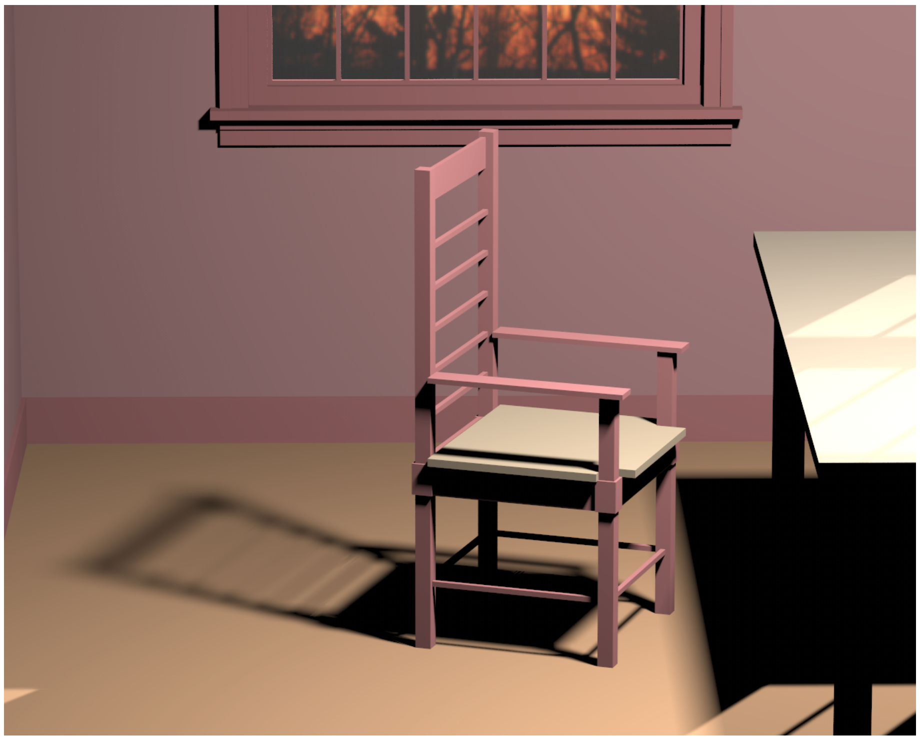
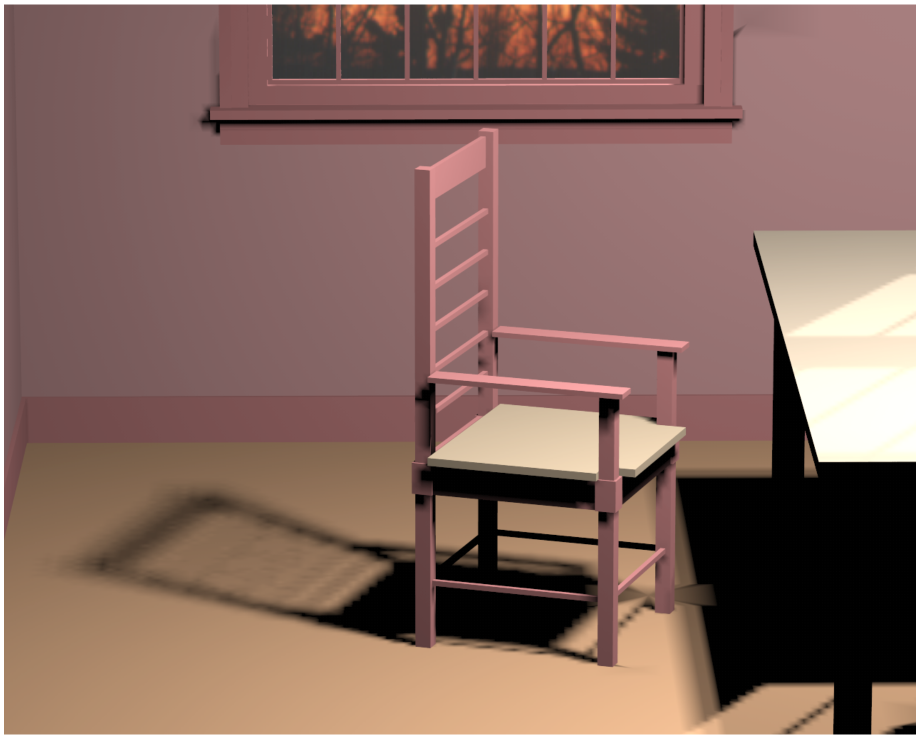

(b)

Figure 14: A closer look at the wall. Image (a) is rendered using a solution produced by discontinuity meshing radiosity. Image (b) is rendered from a standard radiosity solution.



(a)



(b)

Figure 15: A closer look at the chair. Image (a) is rendered using a solution produced by discontinuity meshing radiosity. Image (b) is rendered from a standard radiosity solution.

The model used for the comparison is a simple reproduction of a Victorian-style room consisting of 1382 polygons. The illumination originates from two light sources: the sun, approximated by a polygon placed at a considerable distance, and an area light source at the center of the room's ceiling. Both systems were allowed to perform two iterations, thus the only illumination present in the solutions is the direct illumination from the two light sources. The solution produced by DMR was computed without any need for user intervention. Only an error threshold for adaptive subdivision needed to be specified. The solution consists of 35,540 elements (more statistics are given in Table 1.) Then, a solution with a similar number of elements (37,846) was produced by the SR system. This solution was obtained through a process of trial and error: we experimented with various initial meshing densities and minimum element sizes to obtain reasonable quality. To make a fair comparison no surface-by-surface tweaking was performed; only global meshing parameters were used. For accuracy of the illumination computations, 64 rays per form-factor were used by the SR system.

data type	#
input polygons	1382
patches	1382
BSP tree nodes	3412
BSP tree polygons	5406
final elements	35540

task	time (mins.)	%
build discont. mesh	13	43.3
illumination comp.	15.5	51.7
combining rad. funcs.	1	3.3
miscellaneous	0.5	1.7
total	30	100.0

Table 1: Statistics for the solution computed by the discontinuity meshing radiosity system on an HP720 workstation.

Three images, each with a different view of the model, were rendered from each of the two solutions. The rendering algorithm used was ray-tracing (eye rays only) with adaptive supersampling for anti-aliasing.

A general view of the room is shown in the images in Figure 13. Note how DMR captures the thin shadows from the window on the left wall, many of which are entirely missed by SR. The SR solution also exhibits some shadow leaks from under the window frames.

The next view (Figure 14) is a closer look at the left wall of the room. Note the high quality of the shadows in the image produced from the DMR solution and the realistic softening of the shadows on the wall as we move away from the window. On the other hand, the artifacts in the SR solution are now even more noticeable. The captured shadows are severely aliased: some appear jagged; others are too wide and blurry.

Finally, Figure 15 features a closer view of the chair. Note the high quality of the shadow cast by the chair on the floor. The shadow is sharp and crisp near the legs of the chair, and it becomes softer farther away. In image (b) this shadow exhibits artifacts that were not visible in the general view of the room. Also note that the shadow cast by the armrest on the seat is captured by DMR, but is missing in the SR solution. The fine self shadowing on the chair and the window frame is correctly captured by the new method, yet almost completely absent from the SR solution. Another important difference is visible in the penumbra cast by the table top on the floor. The intensity changes smoothly in (a), while in (b) it appears to change in several steps. This is due to the use of ray-traced form-factors in the SR system, which computes the visibility of the source from each point of the floor by point sampling, in contrast to the accurate hidden surface removal algorithm employed by our method.

The last two views clearly demonstrate the superiority of the new algorithm in computing solutions of high accuracy, independent of the viewer position.

7 Conclusions and Future Directions

We have presented an accurate method for computing the illumination of diffuse polyhedral environments by area light sources and shown how it could be used within a modified progressive radiosity algorithm to solve the diffuse global illumination problem.

While our method is somewhat similar to the recent algorithm by Campbell [14], some important differences exist: In addition to boundaries between umbra, penumbra, and unoccluded regions, other important discontinuities inside the penumbrae are also inserted into the mesh. Thus, a good initial mesh for the penumbra regions is obtained, without resorting to the numerical optimization techniques used in [14]. Additionally, the algorithm does not require convexity of faces in the environment and sources do not have to be constant emitters. Thus, there is no need to split sources into pieces to satisfy these requirements. BSP-trees were extensively used in our implementation; however, they are not inherent to our approach.

Despite the high quality of the images that can be generated, these is still much to be done. We have been able to obtain converged radiosity solutions for simple environments, as well as accurately compute a small number of shots for more complicated environments. However, the great accuracy of our method, which is essential for primary light sources, burdens subsequent iterations as the shooting surfaces become less significant. Less effort should be spent computing contributions which are insignificant with respect to the energy that has already been received. One straightforward solution to that problem would be to stop discontinuity meshing after the first few shots, and approximate the following contributions by collecting to the vertices of a simple grid, using fast ray-traced form-factors. We are currently investigating more sophisticated solutions.

Additional directions for future research include the study of efficient and robust data structures and algorithms for discontinuity meshing and the development of other techniques for improving the performance of the algorithm. Object-space coherence can be exploited to gain considerable speedups. For example, the computation of the contribution from a source to a receiving point can be accelerated by maintaining a list of candidate occluders with every homogeneous region of each receiving surface. A large portion of the execution time is spent calculating wedge-surface intersections. These computations are not dissimilar from ray-surface intersection problems and the considerable body of research done on efficient ray-tracing techniques should be drawn upon to design methods to accelerate the discontinuity meshing algorithm.

Finally, the possibility of extending the new radiosity algorithm to curved surfaces and arbitrary bidirectional reflectance functions should also be investigated.

Acknowledgements

We acknowledge the support of the National Science Foundation under a grant entitled “Interactive Input and Display Techniques” (# CCR-8617880) and the Hewlett-Packard Corporation for generous donations of equipment. The Program of Computer Graphics at Cornell is one of five sites of the new NSF/DARPA Science and Technology Center for Computer Graphics and Scientific Visualization (Grant # ASC-8920219). Matt Hyatt modeled the Victorian room environment used in the simulation.

References

[1] M. F. Cohen and D. P. Greenberg. “The Hemi-Cube: A Radiosity Solution for Complex Environments,” *Computer Graphics*, 19(3), July 1985, pp. 31–40.

[2] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile. “Modeling the Interaction of Light Between Diffuse Surfaces,” *Computer Graphics*, 18(3), July 1984, pp. 213–222.

[3] D. S. Immel, M. F. Cohen, and D. P. Greenberg. “A Radiosity Method for Non-Diffuse Environments,” *Computer Graphics*, 20(4), August 1986, pp. 133–142.

[4] J. R. Wallace, M. F. Cohen, and D. P. Greenberg. “A Two-Pass Solution to the Rendering Equation: A Synthesis of Ray Tracing and Radiosity Methods,” *Computer Graphics*, 21(4), July 1987, pp. 311–320.

[5] F. Sillion and C. Puech. “A General Two-Pass Method Integrating Specular and Diffuse Reflection,” *Computer Graphics*, 23(3), July 1989, pp. 335–344.

[6] F. X. Sillion, J. R. Arvo, S. H. Westin, and D. P. Greenberg. “A Global Illumination Solution for General Reflectance Distributions,” *Computer Graphics*, 25(4), July 1991, pp. 187–196.

[7] M. F. Cohen, S. E. Chen, J. R. Wallace, and D. P. Greenberg. “A Progressive Refinement Approach to Fast Radiosity Image Generation,” *Computer Graphics*, 22(4), August 1988, pp. 75–84.

[8] P. Hanrahan, D. Salzman, and L. Aupperle. “A Rapid Hierarchical Radiosity Algorithm,” *Computer Graphics*, 25(4), July 1991, pp. 197–206.

[9] M. F. Cohen, D. P. Greenberg, and D. S. Immel. “An Efficient Radiosity Approach for Realistic Image Synthesis,” *IEEE Computer Graphics and Applications*, 6(2), March 1986, pp. 26–35.

[10] D. R. Baum, S. Mann, K. P. Smith, and J. M. Winget. "Making Radiosity Usable: Automatic Preprocessing and Meshing Techniques for the Generation of Accurate Radiosity Solutions," *Computer Graphics*, 25(4), July 1991, pp. 51–60.

[11] E. A. Haines. "Ronchamp: A Case Study for Radiosity," SIGGRAPH'91 Frontiers in Rendering Course Notes, July 1991.

[12] F. Tampieri and D. Lischinski. "The Constant Radiosity Assumption Syndrome," in Proceedings of the Second Eurographics Workshop on Rendering (Barcelona, Spain, May 13–15, 1991), May 1991.

[13] A. T. Campbell, III and D. S. Fussell. "Adaptive Mesh Generation for Global Diffuse Illumination," *Computer Graphics*, 24(4), August 1990, pp. 155–164.

[14] A. T. Campbell, III. *Modeling Global Diffuse Illumination for Image Synthesis*, PhD dissertation, University of Texas at Austin, Austin, Texas, December 1991.

[15] P. S. Heckbert. *Simulating Global Illumination Using Adaptive Meshing*, PhD dissertation, Department of EECS, UC Berkeley, California, June 1991.

[16] D. Lischinski, F. Tampieri, and D. P. Greenberg. *Improving Sampling and Reconstruction Techniques for Radiosity*, Technical Report 91-1202, Department of Computer Science, Cornell University, Ithaca, New York, May 1991.

[17] P. S. Heckbert. "Discontinuity Meshing for Radiosity," in Proceedings of the Third Eurographics Workshop on Rendering (Bristol, UK, May 18–20, 1992), May 1992.

[18] J. T. Kajiya. "The Rendering Equation," *Computer Graphics*, 20(4), August 1986, pp. 143–150.

[19] P. Shirley. "Physically Based Lighting Calculations for Computer Graphics: a Modern Perspective," in Proceedings of Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics (Rennes, France, June 11–12, 1990), June 1990, pp. 67–82.

[20] Z. Gigus and J. Malik. "Computing the Aspect Graph for Line Drawings of Polyhedral Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2), February 1990, pp. 113–122.

[21] Z. Gigus, J. Canny, and R. Seidel. "Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6), June 1991, pp. 542–551.

[22] H. Fuchs, Z. M. Kedem, and B. Naylor. "On Visible Surface Generation by a Priori Tree Structures," *Computer Graphics*, 14(3), June 1980, pp. 175–181.

[23] D. Salesin, D. Lischinski, and T. DeRose. "Reconstructing Illumination Functions with Selected Discontinuities," in Proceedings of the Third Eurographics Workshop on Rendering (Bristol, UK, May 18–20, 1992), May 1992, pp. 99–112.

[24] D. R. Baum, H. E. Rushmeier, and J. M. Winget. "Improving Radiosity Solutions Through the Use of Analytically Determined Form-Factors," *Computer Graphics*, 23(3), July 1989, pp. 325–334.

[25] J. R. Wallace, K. A. Elmquist, and E. A. Haines. "A Ray Tracing Algorithm for Progressive Radiosity," *Computer Graphics*, 23(3), July 1989, pp. 315–324.

[26] H. C. Hottel and A. F. Sarofim. *Radiative Transfer*, McGraw-Hill, New York, New York, 1967.

[27] T. Nishita and E. Nakamae. "Half-Tone Representation of 3-D Objects Illuminated by Area Sources or Polyhedron Sources," in Proceedings of The IEEE Computer Society's International Computer Software and Applications Conference (COMPSAC83) (Chicago, Illinois, November 1983), November 1983, pp. 237–241.

[28] M. Rivara. "Numerical Generation of Nested Series of General Triangular Grids," in C. K. Chui, L. L. Schumaker, and F. I. Utreras, editors, *Topics in Multivariate Approximation*, Academic Press, Inc., Orlando, Florida, 1987, pp. 193–206.

[29] B. Trumbore, W. Lytle, and D. P. Greenberg. "A Testbed for Image Synthesis," in Proceedings of Eurographics'91 (Vienna, Austria, September 1–6, 1991), September 1991, pp. 467–480.