Fast and Accurate Hierarchical Radiosity Using Global Visibility

Frédo Durand, George Drettakis and Claude Puech
iMAGIS - GRAVIR / IMAG - INRIA

Recent hierarchical global illumination algorithms permit the generation of images with a high degree of realism. Nonetheless, appropriate refinement of light transfers, high quality meshing and accurate visibility calculation can be challenging tasks. This is particularly true for scenes containing multiple light sources and scenes lit mainly by indirect light. We present solutions to these problems by extending a global visibility data structure, the Visibility Skeleton. This extension allows us to calculate exact point-to-polygon form-factors at vertices created by subdivision. The structure also provides visibility information for all light interactions, allowing intelligent refinement strategies. High-quality meshing is effected based on a perceptually-based ranking strategy which results in appropriate insertions of discontinuity curves into the meshes representing illumination. We introduce a hierarchy of triangulations which allows the generation of a hierarchical radiosity solution using accurate visibility and meshing. Results of our implementation show that our new algorithm produces high quality view-independent lighting solutions for direct illumination, for scenes with multiple lights and also scenes lit mainly by indirect illumination.

Categories and Subject Descriptors: I.3.7 [**Computing Methodologies**]: Computer Graphics—*Three-Dimensional Graphics and Realism*

General Terms: Global Illumination, Global Visibility

Additional Key Words and Phrases: Hierarchical Radiosity, Form Factor Calculation, Discontinuity Meshing, Hierarchical Triangulation, Perception

1. INTRODUCTION AND PREVIOUS WORK

Recent advances in global illumination, such as hierarchical radiosity [Hanrahan et al. 1991] and its combination with discontinuity meshing [Lischinski et al. 1993] have resulted in high quality lighting simulations. These lighting simulations are *view independent* and are suitable for walkthroughs. The quality of the resulting illumination is important everywhere in the scene, since the user can, for example, approach a shadow of an object and see its details.

Despite the high quality of existing techniques, certain aspects of these algorithms are still suboptimal. In particular, deciding when a light-transfer is *refined* appropriately, and thus computed with higher precision is a hard decision; current algorithms ([Hanrahan et al. 1991; Lischinski et al. 1994; Gibson and Hubbold 1996] etc.) include methods based on error bounds which in many cases prove insufficient. Creating a *mesh* to represent lighting variations accurately (notably for shadows) is hard; discontinuity meshing approaches [Lischinski et al. 1993; Drettakis and Sillion 1996] have proposed some solutions for these

iMAGIS is a joint research project of CNRS/INRIA/UJF/INPG.
Address: iMAGIS/GRAVIR, BP 53, F-38041 Grenoble Cedex 09 France
{Fredo.Durand|George.Drettakis|Claude.Puech}@imag.fr
http://www-imagis.imag.fr/

issues which are however often limited in their applicability. Recent approaches (*e.g.*, [Christensen et al. 1996; Ureña and Torres 1997]) avoid this problem by performing a view-dependent, ray-casting "final gather"; view-independence and the capacity for interactive display and walkthroughs are thus sacrificed. Accurate *visibility* calculation is also fundamentally hard, since we have to consider the potential interaction between all polygons in the scene for global illumination.

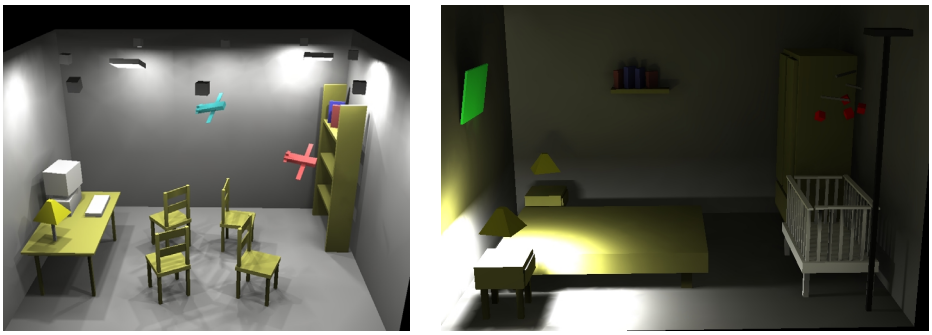
The above three problems, *visibility*, *refinement* and *meshing* are accentuated in the following two lighting configurations: scenes lit by multiple sources and scenes lit mainly by indirect illumination. In this paper we present a new algorithm which addresses the three shortcomings mentioned above. For all three problems, refinement, meshing and visibility previous approaches lack information on accurate *global visibility* relationships in the scene. This information is provided by the *Visibility Skeleton* [Durand et al. 1997]. To achieve our goal, we first extend the Skeleton to provide visibility information at vertices resulting from subdivision of the original input surfaces. The extended Skeleton allows the fast computation of exact point-to-polygon form-factors for any point-polygon pair in the scene. In addition, all visibility information (blockers and all discontinuity surfaces) is available for any polygon-polygon pair.

This global visibility information allows us to develop an intelligent refinement strategy, since we have knowledge of visibility information for *all* light transfers from the outset. We can *rank* discontinuity surfaces between any two hierarchical elements (polygons or patches resulting from their subdivision), using perceptually-based techniques [Gibson and Hubbold 1997]; thus only discontinuities which are visually important are considered. An appropriate mesh is created using these discontinuities; illumination is represented very accurately resulting in high-quality, view-independent meshes. To achieve this in the context of a hierarchical radiosity algorithm, we have introduced a hierarchy of triangulations data structure. Radiosity is gathered and stored at vertices, since the extended Skeleton provides us with the exact vertex-to-polygon form-factor. An appropriate multi-resolution push-pull procedure is introduced. The high-quality mesh, the exact form-factor calculation and the hierarchical triangulation result in lighting simulation with accurate visibility.

Our approach is particularly well-suited for the case of multiple sources since the discontinuity ranking operates simultaneously on *all* light energy arriving at a receiver. Indirect illumination is also handled very well, since visibility information, and thus the refinement and meshing strategies as well as the form-factor computation apply equally well to all interactions, i.e., both direct (from the sources) and indirect (reflected light). Examples of these two cases are shown in Fig. 1. In Fig. 1(a) we see a scene lit by 10 separate light sources, where the multiple shadows are visible but the mesh complexity is reasonable (see Table 2, in Section 7.2). In Fig. 1(b) we see a room lit mainly by indirect lighting; notice the high quality shadows created entirely by indirect light (*e.g.*, on the far wall from the books and lamp).

1.1 Previous Work

The new algorithm we present here is in a certain sense an extension of hierarchical radiosity, using visibility structures, advanced meshing techniques and perceptually-based subdivision. We briefly review hierarchical radiosity methods, accurate visibility techniques and related visibility-based refinement for lighting algorithms and finally perceptually-based refinement for illumination.



(a) (b)

Fig. 1. Images computed using our new hierarchical radiosity algorithm based on the extended Visibility Skeleton and hierarchical triangulations. (a) A scene with multiple sources. The skeleton construction took 2min 23s and the lighting simulation 8min. (b) A scene mainly lit by indirect light. The skeleton construction took 4min 12s and the lighting simulation 6min 58s. Note the shadows caused by indirect illumination, cast by the books on the back wall.

1.1.1 *Hierarchical radiosity* . The hierarchical radiosity algorithm [Hanrahan et al. 1991] allows efficient calculation of global illumination. Lighting calculations are limited to a user-specified level of accuracy, by means of hierarchically subdividing the polygons in the scene into a quadtree, and creating light-transfer "links" at the appropriate levels of the hierarchy. In the original hierarchical radiosity solution [Hanrahan et al. 1991], radiosity is considered constant over each quadtree element. The rectangular nature of the quadtree, and the constant reconstruction result in the need for very fine subdivision for high quality image generation (high quality shadows etc.).

Higher-order (non-constant) methods have also been introduced, notably in the context of wavelet-based solutions [Gortler et al. 1993]. The wavelet-based radiosity solutions presented to date typically operate on discontinuous bases, resulting in visible discontinuities if the solution is displayed directly (*e.g.*, [Christensen et al. 1996]). Zatz [Zatz 1993] used a Galerkin-type method and shadow masks to improve the quality of the shadows generated. To avoid the problem of discontinuous representations the "final gather" step was introduced by [Reichert 1992] and used for wavelet solutions (*e.g.*, [Christensen et al. 1996]). A final gather step consists of creating a ray-cast image, by querying the object-space visibility and lighting information to calculate illumination at each pixel [Ureña and Torres 1997]. This approach allows the generation of high quality images from a coarse lighting simulation, at an additional (frequently high) cost. The solution thus becomes view-dependent, and interactive display and walkthrough capability are lost.

More recently, Bekaert *et al.* have presented an efficient algorithm which combines hierarchical radiosity and Monte-Carlo radiosity [Bekaert et al. 1998]. However, the stochastic nature of the algorithm makes it difficult to refine along shadow boundaries.

1.1.2 *Accurate Visibility and Image Quality.* The accurate calculation of visibility in a lighting simulation is essential: both the numerical quality of the simulation and the visual quality of the resulting image depend on it. The exact computation of visibility between two patches in a scene or between a patch and a point requires the treatment of

visual events. Visual events are caused at boundaries in a scene where the visibility of one object changes with respect to a point of view. Such events occur at visibility boundaries generated by the interaction between vertices and edges of the environment (see Section 2). In the case of the view of a light source, these boundaries correspond to the limits of umbra and penumbra. By choosing certain of these boundaries and using them to guide the (irregular) mesh structure, *discontinuity meshing* lighting algorithms have been introduced resulting in more visually accurate images (*e.g.*, [Heckbert 1992; Lischinski et al. 1992]).

In the vision literature, visual events have been extensively studied [Plantinga and Dyer 1990; Gigus et al. 1991]. The *aspect graph* structure completely encodes all visibility events in a scene. The determination of the visible part of an area light source in computer graphics is exactly the calculation of the aspect of the light at a given point. Algorithms performing this operation by building the complete discontinuity mesh and the *backprojection* data structure (encoding the source aspect) have been presented (*e.g.*, [Teller 1992; Drettakis and Fiume 1994; Stewart and Ghali 1994]). The full discontinuity mesh and backprojection allows the computation of the exact point-to-area form-factor with respect to an area light source. Nonetheless, these methods suffer from numerical problems due to the required intersections between the discontinuity surfaces and the scene polygons, complicated data-structures to represent the highly irregular meshes and excessive computational requirements. The Visibility Complex [Durand et al. 1996] and its simplification, the Visibility Skeleton [Durand et al. 1997], present complete, *global* visibility information between any pair of polygons. We have chosen to use the Visibility Skeleton because of its flexibility, relative robustness (compared to discontinuity meshing) and ease-of-use. A review of necessary machinery from the Skeleton used here is presented in Section 2.

1.2 Visibility-Based Refinement Strategies for Radiosity

In the Hierarchical Radiosity algorithm, mesh subdivision is effected through link refinement. The original algorithm used a *BFV* criterion (radiosity times form-factor modulated by a visibility factor V for partially occluded links). The resulting meshes are often too fine in unoccluded regions, and do not always represent fine shadow details well.

Refinement strategies based on error bounds [Lischinski et al. 1994; Gibson and Hubbold 1996] have improved the quality of the meshes and the simulation compared to the *BFV* criterion. Conservative visibility determination in architectural scenes [Teller and Hanrahan 1993], accurately characterises links as *visible, invisible* or *partially visible*. This triage guides subdivision, allowing finer subdivision in partially illuminated regions.

Discontinuity meshing clearly improves the visual quality of images generated by lighting simulation [Lischinski et al. 1992; Heckbert 1992; Drettakis and Fiume 1994], since the mesh used to represent illumination follows the actual shadow boundaries, instead of finely subdividing a quadtree which attempts to approximate the boundary. One problem is the extremely large number of discontinuities. Tampieri [Tampieri 1993] attempted to limit the number of discontinuity lines inserted, by sampling the illumination along the discontinuities and only inserting those with radiosity values differing more than an predefined threshold. In the context of progressive refinement radiosity, Stuerzlinger [Sturzlinger 1994], only inserted discontinuities at a second level of a regular quadrilateral adaptive subdivision, once a ray-casting step has classified the region as important. The only discontinuities inserted were those due to the blocker identified by the ray caster.

Several methods combining discontinuity meshing with hierarchical radiosity have been presented [Lischinski et al. 1993; Drettakis and Sillion 1996; Hardt and Teller 1996; Boua-

touch and Pattanaik 1995]. Hardt and Teller [Hardt and Teller 1996] present an approach in which potential discontinuities from all surfaces are considered, without actually intersecting them with the blockers. Potential discontinuities are ranked, and those deemed most important are inserted and the lowest level of the quadtree. In [Drettakis and Sillion 1996] the backprojection information is used in the complete discontinuity mesh creating a large number of small triangles. Exact form-factors of the primary source are then computed at the vertices of these triangles. The triangles are then clustered into a hierarchy. Standard [Hanrahan et al. 1991] constant-element hierarchical radiosity follows. The previously cited problems of discontinuity meshing, the expensive clustering step and the fact that the inner nodes of the hierarchy often overlap, limit the applicability of this approach to small models. The only other hierarchical radiosity method with gathering at vertices is that of Martin *et al.* [Martin et al. 1997], which requires a radiosity value at each vertex, and a complex push procedure.

The algorithm of Lischinski *et al.* [Lischinski et al. 1993] is much more complete and relevant to our work. The basis of this approach is to separate the light simulation and rendering steps. This idea is similar in spirit to the use of a "final gather" step. Lischinski *et al.* first compute a "global pass" by creating 2D BSP trees on scene polygons subdivided by choosing important discontinuities exclusively due to the primary sources. The 2D BSP tree often incurs long splits and consequently long or thin triangles, which are inappropriate for high quality lighting simulation. The second, view independent, "local pass" recomputes illumination at the vertices of a triangulated subdivision of the leaf elements of the BSP tree. To achieve high quality images, the cost of triangulation and shading (light recomputation at vertices using "method D" [Lischinski et al. 1993]), is higher than that of the actual lighting simulation (if we ignore the initial linking step).

1.3 Perceptually Based Refinement

Recently, perceptually-based error metrics have been used to reduce the number of elements required to accurately represent illumination (*e.g.*, [Gibson and Hubbold 1997; Hedley et al. 1997]). Tone-reproduction approaches [Tumblin and Rushmeier 1993; Ward 1994] are used to map calculated radiosity values to display values which convey a perceptual effect closer to that perceived by a real world viewer. Since display devices have limited dynamic range compared to real world luminance values, the choice of this mapping is very important. The tone reproduction mappings of [Tumblin and Rushmeier 1993; Ward 1994] depend on two parameters: a world adaptation level which corresponds loosely to the brightness level at which a hypothetical observer's eye has adapted, and a display adaptation level which corresponds to the brightness displayed on the screen. Choice of these parameters affects what will be displayed, and, more importantly, which differences in radiosities will actually be perceptible in the final image. Most notably, one can define a "just noticeable difference" using this mapping. In the context of lighting, a just noticeable difference would correspond to the smallest difference in radiosity values, which once transformed via tone reproduction, will be visible to the viewer of the display. Display adaptation is typically a fixed value (*e.g.*, half the maximum display luminance [Ward 1994]), while the world adaptation level can be chosen in a number of different ways. Using static adaptation [Gibson and Hubbold 1997], one uses an average which is independent of where the observer is looking, while dynamic adaptation (which is closer to reality) changes depending on where the observer is looking. Gibson and Hubbold [Gibson and Hubbold 1997] use tone reproduction to guide subdivision in a progressive refinement

radiosity approach, thus allowing a subdivision only if the result will be “just noticeable”.

Hedley *et al.* [Hedley et al. 1997], in a similar spirit, use a tone mapping operator to determine whether a discontinuity should be inserted into a lighting simulation mesh. This is performed by sampling across the discontinuity (in a manner similar to that of Tampieri [Tampieri 1993]), but also orthogonally across the discontinuities. This results in an important reduction of discontinuities without loss of visual quality.

1.4 Paper Overview

Previous algorithms surveyed above provide view-independent lighting simulations which are acceptable for many situations. In particular, quadtree based hierarchical radiosity provides fast solutions of moderate quality, even for scenes mainly lit indirectly. Nonetheless, in walkthroughs the observer often approaches regions of shadow, and in these cases the lack of shadow precision is objectionable. Previous approaches based on discontinuity meshing alleviate this problem for direct lighting, but rapidly become impractical for scenes with many lights, or for which indirect lighting is dominant. Their limitations are due to the sheer number of discontinuity surfaces that need to be considered when computing indirect illumination and the complexity of the meshes which result. These issues are discussed in [Lischinski et al. 1992] and [Tampieri 1993]. The solutions adopted to date have restricted the use of discontinuity information to those from primary light sources [Lischinski et al. 1993; Drettakis and Sillion 1996]; for subsequent light bounces (secondary, tertiary etc.), approximate ray-casting approaches are used for visibility computations in light transfer.

The new algorithm presented here allows the generation of accurate shadows for a more general class of scenes, including those with dominant indirect illumination. To achieve this goal we extend the Visibility Skeleton to support view calculation at vertices resulting from subdivision, and to use a link-based storage mechanism which is more adapted to a hierarchical radiosity approach. This extended structure is presented in Section 2. The resulting structure allows us to select and insert discontinuity lines for all light transfers, and to calculate exact point-to-area form-factors rapidly, using the visibility information provided. These choices required us to develop a new hierarchical radiosity algorithm, with gathering at vertices, based on embedded *hierarchical triangulations* allowing the mesh to follow discontinuity lines. The details of this structure, and the novel push-pull algorithm are presented in Section 3. In Section 4 we present the new hierarchical radiosity algorithm using accurate global visibility and we present the new point-polygon and polygon-polygon link data structures. In Section 5 we present the corresponding refinement processes and the visibility updates required for their use. In Section 6 polygon subdivision and the perceptually-based refinement criterion are described. We then present results of our implementation, as well as a discussion of relative limitations and advantages of our approach, and we conclude.

2. THE VISIBILITY SKELETON

The Visibility Skeleton [Durand et al. 1997] is a data structure encoding all the global visibility relationships in a 3D scene. It is based on the notion of *visibility events*.

A visibility event is the locus of a topological change in visibility. An example is shown in Fig. 2(a) (taken from [Durand et al. 1997]). When the viewpoint moves from left to right, vertex *v* as seen from the observer will no longer lie on the floor, and will now be on the polygon adjacent to edge *e*. We say that there is a topological change in the view from

the eyepoint.

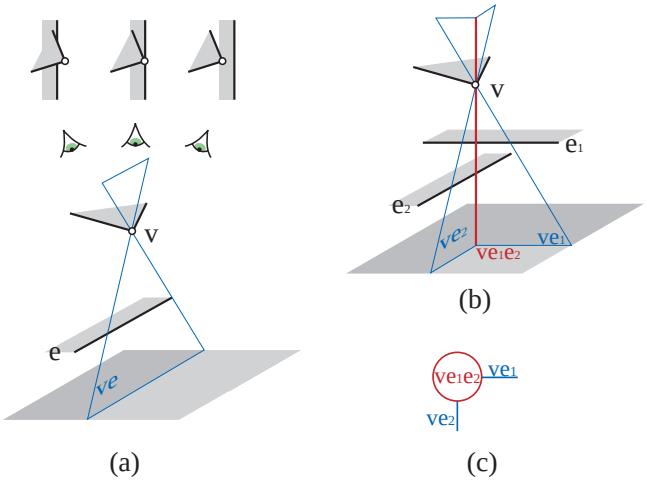


Fig. 2. (a) An EV line swath, (b) The VEE node is adjacent to two line swaths (c) The graph structure induced (Figure taken from Durand *et al.* [Durand et al. 1997])

A visibility event is a 1D set of lines: in Fig. 2(a) the EV event is the set of lines going through v and e; it can be parameterized by the abscissa on e. We call the surfaces swept by such sets of lines *line swaths*. Such swaths are caused by the interaction of an edge and a vertex (EV swaths) or three edges (EEE) swaths.

The extremities of these 1D line sets are lines with no degrees of freedom: the *extremal stabbing lines*. These are lines passing through four edges of the scene. Examples are vertex-vertex (VV) lines passing through two vertices, vertex-edge-edge (VEE) lines passing through two edges and a vertex (see for example Figure 2(b)) or 4-edge ($E4$) lines passing through four edges.

This construction naturally defines a graph in line-space. The nodes are the extremal stabbing lines and the arcs are the line swaths. The nodes of the graph are adjacent to a certain number of arcs (swaths), which are defined in a catalogue for each type of node [Durand et al. 1997]. Fig. 2(b) shows the adjacencies of a VEE extremal stabbing line and two EV critical line swaths. The graph structure induced, consisting of a node and the two arcs, is shown in Fig. 2(c).

Efficient access to the visibility information is provided by means of an n^2 array (where n is the number of objects in the scene) indexed by the polygons at the extremities of the swaths: A cell of the array indexed by polygons (P,Q) stores in a search tree all the line swaths whose extremities lie on P and Q.

The *visibility skeleton* is the graph of line swaths and extremal stabbing lines together with the array of search trees.

The construction of the Skeleton proceeds as a set of nested loops over the edges and vertices of the scene to determine the nodes (extremal stabbing lines). First simple cases (*e.g.*, VV) are found, and subsequent loops over the edges allow the identification of VEE and $E4$ nodes.

Once a potential extremal stabbing line is detected, it is tested for occlusion using ray-casting. If an object lies between its generators, it is discarded. Otherwise a node is created. The ray-casting operation also provides the extremities of the node.

Once a node is created, its neighbourhood in the graph is updated using the catalogue of adjacent arcs. If an adjacent arc has already been created (because of its other extremity) it is just linked to the new node; otherwise it is created and linked. The array of search trees is used for efficient search of the existing arcs.

It is important to note that the line swaths are not actually constructed geometrically: only the extremal stabbing lines are involved in the geometric construction. This makes the algorithm more robust, since only ray-casting is needed, as opposed to traditional discontinuity meshing which requires complicated swath-polygon intersections [Drettakis and Fiume 1994].

2.1 Extensions to the Visibility Skeleton

2.1.1 *Memory requirements.* The storage of the arcs of the Skeleton in a two dimensional array incurs an $O(n^2)$ cost in memory. In the scenes presented in [Durand et al. 1997] half of the memory was used for the array, in which more than 95% of the cells were empty! It is even more problematic when the scene is highly occluded such as in the case of a building where each room sees a only fixed number of other rooms: the number of arcs is only $O(n)$. Moreover, for our lighting simulation, we will need to subdivide the initial polygons into sub-patches and incrementally compute visibility information between some pairs of sub-patches, but not all.

For these reasons we store the set of critical line swaths between two polygons on the polygons themselves. Each polygon P stores a balanced binary tree; each node of this tree contains the set of arcs between P and another polygon Q. This set is itself organized in a search tree (see Figure 3). Each set of arcs is referenced twice, once on P and once on Q. In the same manner, each vertex V has a search tree containing the sets of arcs between V and a polygon Q (which represents the view of Q from V). These sets of arcs are closely related to the notion of *links* in hierarchical radiosity as we will see in Section 4.2.

For the scenes presented [Durand et al. 1997], this approach results in an average memory saving of about 30%. Moreover, we have ran our modified version of the visibility skeleton on a set of scenes consisting of a room replicated 2, 4, and 8 times, showing roughly linear memory growth for the skeleton. Using a binary tree instead of an array incurs an additional $O(\log n)$ time access cost, but this was not noticeable in our tests.

2.1.2 *Visibility Information at Vertices from Subdivision.* To permit the subdivision of surfaces required to represent visual detail (shadows etc.) on scene polygons, visibility skeleton information must be calculated on the triangles created by the subdivision and the corresponding interior vertices. The process is presented in detail in Section 5.

Since the visibility information is now stored on polygons and vertices (instead of in a 2D array), the generalization to subdivided polygons is straightforward. On each sub-patch or sub-vertex, we store the visibility information only for the patches it interacts with.

2.2 Treating Degenerate Configurations

Computational geometry often makes the assumption that the scenes considered are in a "general configuration". Unfortunately, computer graphics scenes are very often highly degenerate: many points are aligned, segments or faces are parallel or coplanar and objects

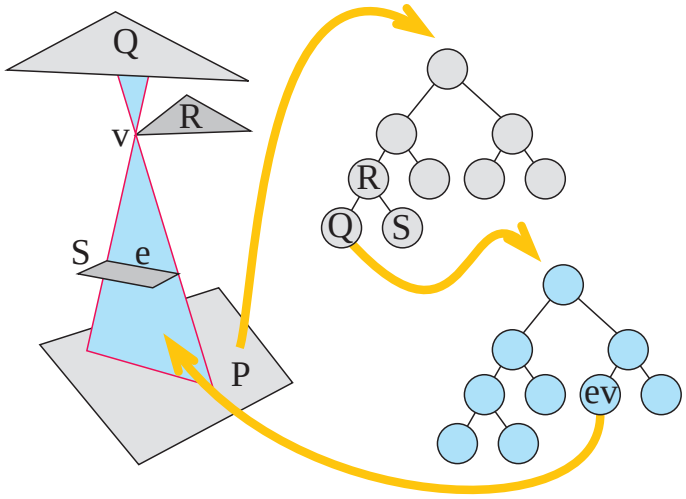


Fig. 3. Summary of the visibility skeleton structure. Each polygon stores a search tree indexed by the polygons it can see. For each pair of polygons, a search tree of visibility events is stored.

touch each other.

This results in degenerate visibility events; *e.g.*, *VVV* extremal stabbing lines passing through three aligned vertices, or $E5$ stabbing lines going through five edges.

These degenerate configuration cause duplicate line swaths and result in numerical instabilities in the occlusion test of a potential extremal stabbing line. This line may then be randomly discarded. Inconsistencies can thus appear in the neighbourhood of the corresponding nodes of the graph. A consistent policy has to be chosen to include these nodes and their adjacent arcs or not.

We first have to identify the occurrence of these problems. When a potential extremal stabbing line is tested for occlusion, we also check for grazing objects. This requires a simple modification to the point-in-polygon test used for the ray-casting occlusion test of the potential extremal stabbing lines. We thus detect the intersection with a silhouette edge or vertex.

We also have to deal with the aforementioned degenerate extremal stabbing lines. A first possibility is to explicitly create a catalogue of all these degeneracies. This approach however quickly makes the implementation intractable because of the large number of different cases. We have chosen to always consider the simplest configuration, that is the one in which we have the smallest number of visual events. For example, if four edges E_1, E_2, E_3 and E_4 are parallel in that order, we consider that E_2 occludes E_1 and then E_3 occludes E_2 etc. The configurations to be treated are thus simpler and correspond to the standard Skeleton catalogue of events. The problems of numerical precision are treated using a consistent ε threshold for equality and zero tests.

3. IRREGULAR HIERARCHICAL TRIANGULATIONS FOR ILLUMINATION

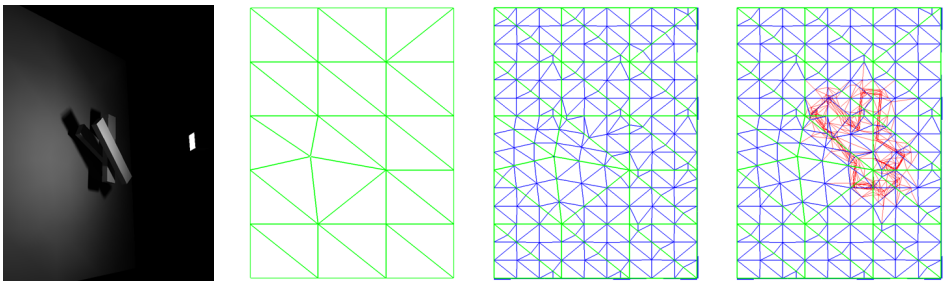
In previous work (*e.g.*, [Heckbert 1992; Lischinski et al. 1992]) it has been shown that the creation of a mesh well adapted to the discontinuities in illumination results in images

of high visual quality. Incorporating such irregular meshes into a hierarchical radiosity algorithm presents an important challenge. As mentioned in Section 1.1.2, most previous algorithms [Lischinski et al. 1993; Drettakis and Sillion 1996] addressing this issue have restricted the treatment of discontinuities to those due to direct (primary) illumination.

The core of the problem is that two conflicting goals are being addressed: that of a simple regular hierarchy, permitting straightforward manipulations and neighbor finding and that of an essentially irregular mesh, required to represent the discontinuity information. The first goal is typically achieved using a traditional quadtree structure [Hanrahan et al. 1991] and the second typically by a BSP-type approach [Lischinski et al. 1993].

In previous approaches, discontinuity information and accurate visibility were incorporated into constant-element hierarchical radiosity algorithms. In the case of the Skeleton, this would be wasteful, since we have all the necessary information to compute *exact* form-factor from any polygon in the scene to any vertex (see Section 5 to see how this is also true for vertices resulting from subdivision). Gathering to vertices introduces one important complication: contrary to elements whose level in the hierarchy is clearly defined, vertices are shared between hierarchy levels.

As a solution to the above issues, we introduce hierarchical triangulations for hierarchical radiosity. Our approach has two major advantages over previous hierarchical radiosity methods: (i) it adapts well to completely irregular meshes and this in a local fashion (triangulations contained in triangulations), avoiding the artifacts produced by splitting edges of a 2D BSP tree and (ii) it allows gathering to vertices by a "lazy wavelets"-type (or subsampling) construction (see the book by Stollnitz *et al.* [Stollnitz et al. 1996] pp 102–104 and 152–154). It preserves a linear approximation to radiosity during the gather and the push process of the solution.



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

Fig. 4. Hierarchical Triangulation Construction. Notice how the triangles are overall well shaped but also well adapted to local detail. (a) Scene geometry: the leftmost polygon is illuminated by the area source on the right pointing leftwards. (b) First level of subdivision for the leftmost polygon (green). (c) Second level (blue). (d) third level (red).

3.1 Hierarchical Triangulation Construction

Our hierarchical triangulation construction has been inspired by that of de Floriani and Puppo [De Floriani and Puppo 1995]. As in their work we start with an initial triangulation, which is a constrained Delaunay triangulation (CDT). The CDT allows the insertion of constrained edges into the triangulation, which are not modified to satisfy the Delaunay

property and thus remain “as is”. Each triangle of the initial triangulation can be subdivided into a sub-triangulation, and so on recursively. At each level, a CDT is maintained.

An example of such a construction is shown in Figure 4, clearly showing the first advantage mentioned above. As we can see, the triangulation maintains well-shaped triangles everywhere in the plane, while providing fine details in the regions where this is necessary. The representation of such detail induces irregular subdivision at the finer levels.

Our hierarchical triangulation is “matching”, in the sense that edges split across two levels of a triangulation are done so at the same point on the edge. At the end of each subdivision step an “anchoring” operation is performed by adding the missing points in the neighboring triangles, thus resulting in a conforming triangulation across levels required for the push phase of hierarchical radiosity.

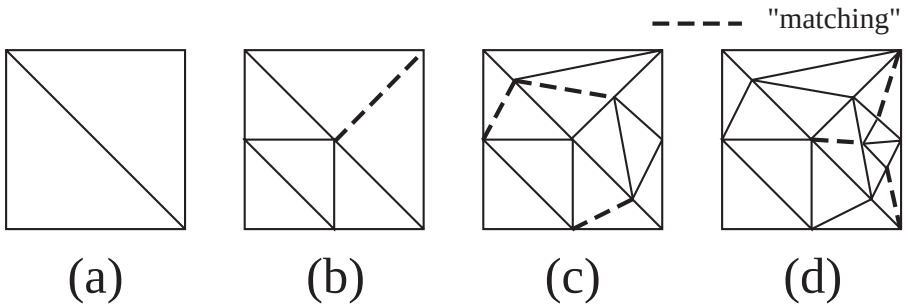


Fig. 5. The “matching” constraint for the Hierarchical Triangulation. The sequence shows subsequent segment insertions. The dashed lines show the insertions performed to enforce the “matching” constraint.

As mentioned above, vertices are shared between different levels of the triangulation. The initial level of a triangulation is an *HPolygon*, which contains an *HTriangulation* child once subdivided, where the prefix *H* represents the hierarchical nature of the construction.

To transmit neighborhood information between levels (for the matching operation), we use a special *HEdge* structure. An *HEdge* is shared between hierarchy levels by all edges which correspond to the same segment. It contains pointers to sub *HEdge*’s when it is subdivided. To perform a matching operation we determine whether the edge on which we insert a point *p* has already been split. We then add the new points corresponding to the previously split vertices, and split the *HEdge* at the point *p*. The neighbouring triangle can thus identify the newly inserted sub-*HEdge*’s from the shared *HEdge*. For example, in Figure 5, after the subdivision of the lower left triangle in Fig. 5(a), the *HEdge* shared between the two triangles notifies the upper right triangle that the edge has been split, and facilitates the matching operation as shown in Fig. 5(b).

3.2 Linear Reconstruction of Illumination using Hierarchical Triangles

The second advantage, that of linear reconstruction of illumination across irregular meshes, requires the use of a “lazy-wavelet” or “sub-sampling” type construction. Lazy wavelets provide an elegant formalism for a simple approach: a piecewise linear approximation is refined through the addition of new sampling points [Stollnitz et al. 1996].

As mentioned above, in our hierarchical triangulation representation of radiosity, vertices will be shared between hierarchy levels. As a consequence, traditional push-pull procedures [Hanrahan et al. 1991] cannot be directly applied.

To understand why, consider the 1D example shown in Fig. 6. Segment $v_a v_b$ is illuminated by two light sources S_1 and S_2. Assume that initially both light transfers are refined, and vertex v_1 is added. This results in the configuration of level 1 (Fig. 6). The light transfer with S_2 is further refined with the addition of v_2 on the right, thus splitting segment $v_1 v_b$. Finally, the light transfer from S_1 is refined on the left, with the addition of v_3.

To determine the light contribution of S_1 in the interval $[v_1, v_b]$ we interpolate between the values transfered by $S_1 \to v_1$ and $S_1 \to v_b$, which are "represented" at level 1. However, for light S_2, we must interpolate in the subinterval $[v_1, v_2]$ using the transfers determined by $S_2 \to v_1$ and $S_2 \to v_2$, and in the subinterval $[v_2, v_b]$ using the values determined by $S_2 \to v_2, S_2 \to v_b$, all of which are "represented" at level 2. Thus v_1 is shared between level 1 and level 2. As a consequence traditional push-pull procedures with gathering at elements rather than vertices cannot work, since they require that an element clearly belong to a certain hierarchy level.

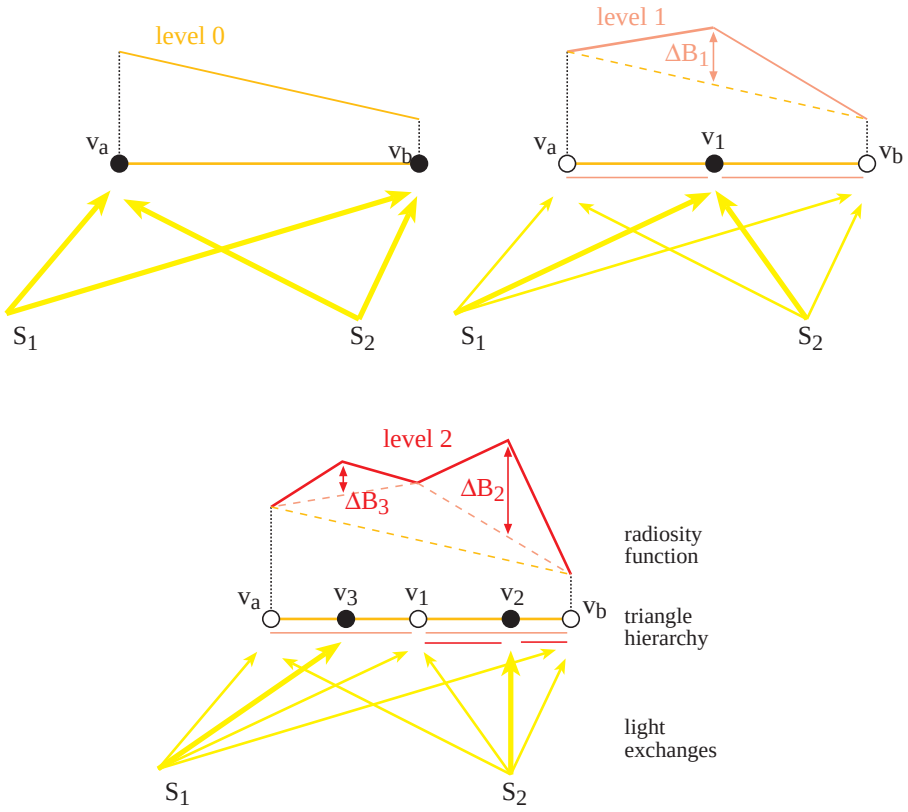


Fig. 6. Consistent multiresolution representation with lazy wavelets. Instead of storing radiosity values, we store the difference of the radiosity values at refined vertices.

A naive solution would be to duplicate vertex v_1 to differentiate exchanges simulated at different levels of the hierarchy. This however is not sufficient, since it is unclear how to perform the push operation. In particular, assume that we had one representation of v_1 for level 1 and one for level 2. It is unclear where the transfers $S_1 \rightarrow v_1$ and $S_2 \rightarrow v_1$ should be stored. If $S_1 \rightarrow v_1$ is stored at level 1, we can interpolate correctly in the interval $[v_1, v_b]$ to perform the push onto vertex v_2. However the value will no longer be available at level 2 to enable the interpolation between v_3 and v_1. In a symmetrical manner, we need $S_2 \rightarrow v_1$ to perform the interpolation at level 1 for the interval $[v_a, v_1]$ and the push on v_3, and at level 2 for the interpolation in the interval $[v_1, v_2]$. With gathering at vertices and linear interpolation, it no longer makes sense to speak of a transfer at a given level of the hierarchy.

We use lazy wavelets to provide a solution to these problems. Instead of storing the actual radiosity value, at refined vertices we store the *radiosity difference* as shown in Fig. 6. This is the difference between the radiosity value at the current level and the interpolated value of the immediate ancestor. This provides a multi-resolution representation, since certain light transfers are refined more in the appropriate regions with the addition of new links.

The push procedure is then straightforward: To compute the total radiosity at a vertex, we interpolate the value of its ancestor, and add the radiosity difference. We obtain the total value at this vertex, which is thus recursively pushed down the hierarchy in a breadth-first manner.

This construction is directly applicable to the 2D case, by using barycentric coordinates (or bilinear for quadrilaterals) for the interpolation. We thus can simply perform a push operation on a hierarchical triangulation with gathering at the vertices.

Note however that it is slightly more involved to compute the difference of a light transfer than the total light transfer. Section 4.2.2 will deal with this problem through the use of "negative" links.

The pull computation is simpler, since we pull values to the triangles. At each triangle leaf, the value given is simply the average of values at the vertices (after the push). An intermediate node receives as a value the area-weighted average of its children triangles, as in standard hierarchical radiosity.

The advantages of this approach are that we can now create a consistent multi-resolution representation of radiosity over the hierarchical triangulation, while gathering at vertices. In addition, the push operation maintains a linear reconstruction of the radiosity function down to the leaf level.

4. VISIBILITY-DRIVEN HIERARCHICAL RADIOSITY: ALGORITHM AND DATA STRUCTURES

The hierarchical triangulation structure is one of the tools required to effect visibility-driven hierarchical radiosity. In particular, we can efficiently represent the irregular lighting discontinuities in a hierarchical structure. In addition, the information contained in the extended Visibility Skeleton provides *exact* and *global* visibility information. As a consequence, we can compute exact (analytical) area-to-point form factors for any light transfer, direct (primary) or indirect. The information contained in the Skeleton arcs (i.e. the visibility events affecting any light transfer) also allows the development of intelligent refinement criteria, again for any exchange of light.

In what follows we present our new algorithm which uses the extended Skeleton and the

hierarchical triangulations for efficient refinement and accurate light transfer.

4.1 Algorithm Outline

Our new algorithm is outlined in Fig. 7. It begins with the creation of the Visibility Skeleton for the given scene, using the improved link-based approach (Section 2.1.1). After this step, we have all the information available to calculate form-factors from each polygon to each (initial model) vertex in the scene. In addition, polygon-polygon visibility relationships are available directly from the skeleton, thus obviating the need for initial linking (i.e. only necessary links are created). After computing the form-factors of the initial polygons to the initial vertices, a "gather" step is performed to the vertices, followed by a "push-pull" process. In practice we perform a fixed number of iterations; however it would be possible to iterate to convergence, since these iterations are not computationally expensive.

Note that even at this very initial phase, the form-factors at the vertices of the scene are exact. To bootstrap subdivision, we first insert the maxima of the light source illumination functions into large receiver polygons (procedure *insertMaxima()*, see also Section 6.2).

```
visibilityDrivenHR
{
   computeSkeleton()          // compute the Visibility Skeleton
   computeCoarseLighting()  // 3 gather push-pull
   insertMaxima()             // insert the maxima of light sources into meshes
   while( !converged() ) do
      subdividePolygons()    // Refine the polygons using visibility info
      refineLinks()             // Refine the links using visibility info
      gatherAtVertices()      // Gather at the vertices of the Hierarchical Triangulation
      pushPull()
   endwhile
}
```

Fig. 7. Visibility Driven Hierarchical Radiosity

Once the system has been initialized in this manner, we begin discontinuity based subdivision (*subdividePolygons()*) and link refinement (*refineLinks()*). Using the global visibility information, we are capable of subdividing surfaces by following "important" discontinuities. After the completion of each subdivision/refinement step, a gather/push-pull operation is performed, resulting in a consistent multi-resolution representation of light in the scene.

In the following discussion we use the terms "source" and "receiver" for clarity. A source is any polygon in the scene which emits or reflects light. For secondary or tertiary illumination, for example, "sources" will be polygons other than the primary light sources (*e.g.*, the walls, ceiling or floor of a room).

In the rest of this section, we present the link data structures and discuss issues related to form-factor calculation and multi-resolution link representation. In Section 5 we describe the refinement process for links, and the details of visibility updates; in Section 6 we

present the polygon subdivision strategy and the perceptually-based refinement criterion used to effectively perform the subdivision.

4.2 Link Data Structures and Form-Factors

The central data structures used for our lighting solution are the links used to perform subdivision and light transfers. In contrast to previous hierarchical radiosity methods, two distinct link types are defined: point-polygon links which are used to gather illumination at vertices, and polygon-polygon links, which are used to make refinement decisions and to maintain visibility information while subdividing.

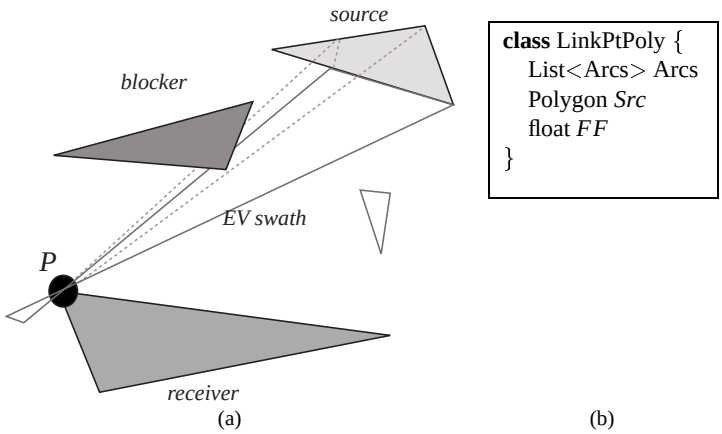


Fig. 8. (a) A point-polygon link used to gather illumination at vertex *P*. Note that all the arcs of the skeleton between *P* and the polygon *source* are stored with the link, *e.g.*, the *EV* swath shown. (b) The corresponding data structure.

4.2.1 *Point-Polygon Links.* As mentioned above, the skeleton provides all the information required to calculate the exact area-to-point form-factor from any polygon in the scene to any vertex. By updating the view information as shall be discussed below (Section 5.3), we extend this capacity to new vertices created by subdivision.

There are numerous advantages to calculating illumination at vertices. When computing radiosity at patch centers, the result can be displayed as flat shaded polygons. To provide a more visually pleasing result, the radiosity values are usually first *extrapolated* to the patch vertices and then interpolated. Inevitably, this introduces many artifacts in the approximation of the original radiosity function. In addition, it is much cheaper and simpler to compute exact polygon-to-vertex form-factors than polygon-to-polygon form-factors. Computing radiosity at vertices was first introduced by Wallace *et al.* [Wallace et al. 1989] in the context of progressive refinement radiosity. For hierarchical radiosity, the fact that vertices can be shared between different levels renders gathering at vertices more complicated.

A point-polygon link and the corresponding data structure are shown in Fig. 8. The point-polygon links are stored at each vertex of the hierarchical triangulations. A point-polygon link stores the form-factor calculated, as well as the arcs of the visibility skeleton

(visibility events of the view) between the point and the polygon. An example is shown in Fig. 8, where the *EV* swath is stored with the link between point *P* and the source polygon.

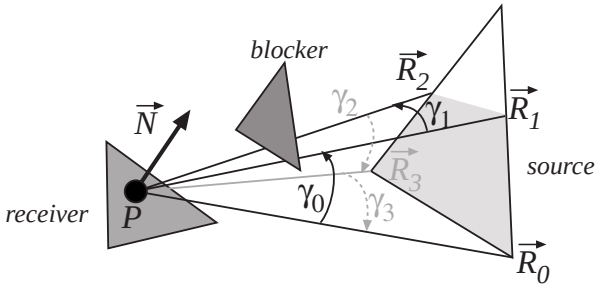


Fig. 9. Geometry for the calculation of a form factor

The point-area form factor is computed analytically using the formula in *e.g.* [Baum et al. 1989]. Consider Fig. 9.

$$F_{P,source} = \frac{1}{2\pi}\vec{N} \bullet \sum \gamma_i \frac{\vec{R}_i \times \vec{R}_{i+1}}{\| \vec{R}_i \times \vec{R}_{i+1} \|}$$

The sum is evaluated using the arcs of the skeleton stored in the point-polygon link. $\vec{R}_i$ and $\vec{R}_{i+1}$ correspond to the two nodes (extremal stabbing lines) of the arc.

Fig 10 shows an example of form-factor computation with the Visibility Skeleton; the computation is exact. For comparison, the average (relative) error is given using ray-casting and a jittered grid sampling on the source (both the kernel and visibility are evaluated by Monte-Carlo). Note that 36 rays are needed to have a mean error of 10%; numerical error on the form-factor is being measured. As expected from stratified sampling the convergence rate is about $O(n^{-\frac{3}{4}})$ [Mitchell 1996], since the function to be integrated is only piecewise continuous because of the visibility term. In Section 7.2.1 we will show the effect of this accuracy on the image quality.

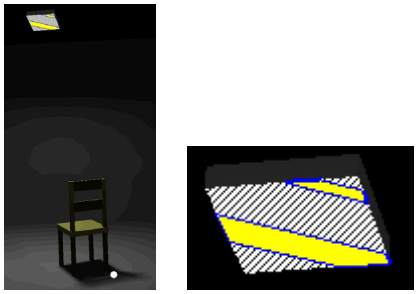
4.2.2 *Multi-Resolution Link Representation.* To maintain the multi-resolution representation of radiosity in the hierarchical triangulation, we require the representation of ΔB as described in Section 3.2, for the push phase of the push-pull procedure.

When a new vertex is inserted into a receiver polygon, "negative links" are created, from the source to the three vertices of the triangle containing the newly inserted vertex [†]. These links allow the direct computation of ΔB as follows:

$$\Delta B \;=\; B_l - \sum_{i=0..2} c_i \, B^i_{nl}, \tag{1}$$

where B_l is the radiosity gathered from the positive link, B^i_{nl} is the radiosity gathered from the negative links and c_i are the barycentric coordinates of vertex P_i. An example of negative links is shown in Fig. 11(b).

[†]In practice, these are simply pointers to the previously existing links.



	Skeleton	4 rays	16 rays	36 rays	64 rays	100 rays
time	0.07ms	0.5ms	1.7ms	3.8ms	6.7ms	10.4ms
error	0	50%	20 %	9.6%	7.6%	4.6 %

Fig. 10. Example of Form-Factor computation from the white point on the floor to the area light source using the visibility skeleton and ray-casting with jittered sampling. The hidden part of the source is hatched. The Visibility skeleton timing does not include the visibility update (about 0.13ms per link on average for this image).

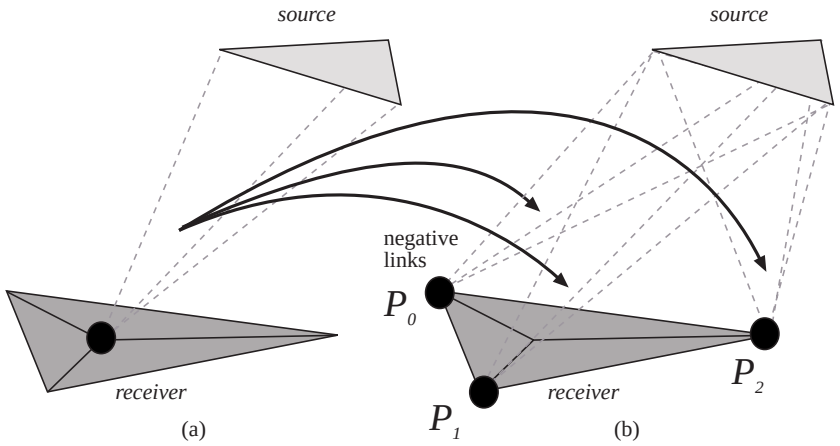


Fig. 11. (a) A newly inserted point (in black) and the point-polygon link to the source; the vertex points to the (b) three new negative links to the source used for the ΔB representation.

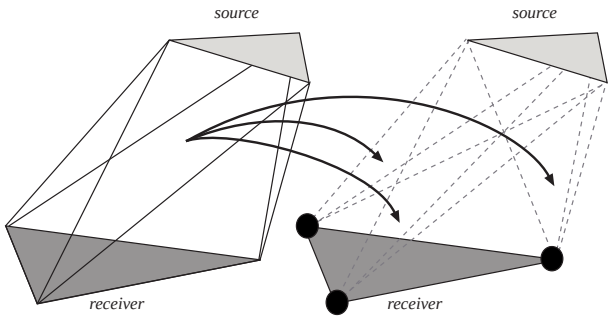
The entire gather/push-pull process is illustrated in Fig. 12. Note that the field *child* of an *HPolygon* is the associated triangulation.

4.2.3 *Polygon-Polygon Links.* The polygon-polygon link is used mainly to determine how well the light transfer is represented, in a manner similar to that of the links in previous hierarchical radiosity algorithms. This information is subsequently used in the refinement process as described below.

A polygon-polygon link stores visibility information via pointers to the point-polygon links (two sets of three links for a polygons pair) between each polygon and the vertices of the other polygon. A polygon-polygon link is illustrated in Fig. 13, with the corresponding point-polygon links from the source to the receiver.

```
Gather ()
{
   for each vertex v
      ΔB_v = gather(positive links_v) - gather(negative links_v)
}
Push (HPolygon poly)
{
   if child(poly) == NULL return
   for each vertex v in triangulation child(poly)
      B_v = ΔB_v+ interpolation (poly, v)
   for each triangle t in triangulation child(poly)
      Push(t)
}
Pull (HPolygon poly)
{
   if child(poly) == NULL
      B_poly = average of B_v, for all v vertex of poly
      return
   for each triangle t in triangulation child(poly)
      Pull(t)
   B_poly =average of B_t, for all t in child(poly)
}
```

Fig. 12. Gather and push-pull




```
class LinkPolyPoly {
   List<LinkPtPoly> ptPolyLinks
   Polygon Src
}
```

(c)

Fig. 13. (a) A polygon-polygon link used to estimate illumination transfer between two polygons (b) The polygon-polygon links store pointers to the 6 corresponding point-polygon links: 3 of them (*source* → *receiver*) are shown here (c) The corresponding data structure.

Note that in the case of a subdivided polygon, all the neighboring triangles of a vertex *v* share all the point-polygon links related to *v*.

5. LINK REFINEMENT

Now that the link data structures have been described in detail, we can present the link refinement algorithm. Note that the process is slightly more involved than in the case of

standard hierarchical radiosity (*e.g.*, [Hanrahan et al. 1991]), because the subdivision is not regular and the existence of the two link types requires some care to ensure that all updates are performed correctly. This section describes how the links are actually refined.

5.1 Refinement overview

Consider a light exchange from a source polygon to a receiver polygon. Because we gather radiosity from the polygon at the vertices, two kinds of refinement can be necessary.

—*Source-refinement* if the radiosity variation over the source polygon is too high,

—*Receiver-refinement* if the sampling on the receiver is too coarse.

The link refinement algorithm is straightforward: for each polygon and each of its polygon-polygon links, the link is tested for refinement. If the test fails, the link is refined and the new point-polygon and polygon-polygon links are created. Finally, the visibility of the link is updated. The refinement test uses a perceptually-based refinement criterion, based on the visibility information contained in the extended Skeleton (see Section 6.4). Note that since we compute exact point-to-area form factors, the source refinement cannot be caused by the inaccuracy of the form-factor computation. It can only happen because the radiosity of the source is not uniform, i.e., if a receiver-refinement has occured on the source in another exchange.

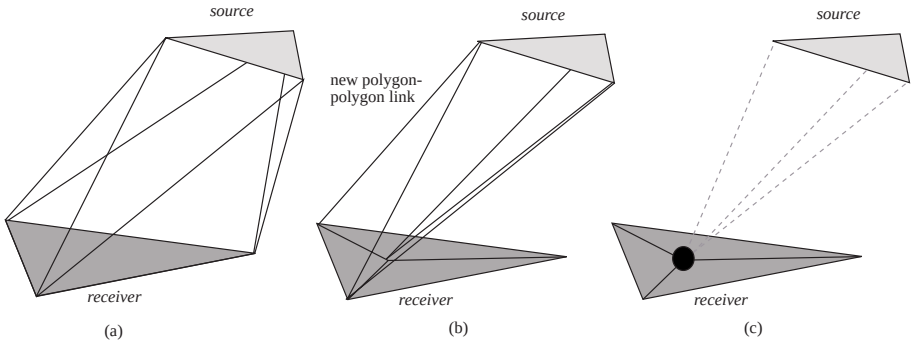


Fig. 14. Receiver refinement: (a) Original polygon-polygon link (b) Insertion of a point on the receiver and one of the three new polygon-polygon links created (c) The additional point-polygon link to the source.

5.2 Source and Receiver Refinement

The first type of refinement is that of a source. If the representation of radiosity across the source is considered insufficient for the given transfer (i.e., the variation of radiosity is too high across the source), the link will be refined. Note that the geometric subdivision of the source has occured at a previous iteration, typically due to shadowing. New polygon-polygon links are created between the original receiver and the sub-triangles of the source. New point-polygon links are created for each vertex and each source sub-triangle, and the corresponding visibility data is correctly updated (see Section 5.3).

The second type of refinement is that of the receiver. For example, in Fig. 14, a point is added to the receiver. As a consequence, the triangulation is updated and three new

polygon-polygon links are added. One of these is shown in Fig. 14(b). In addition, a new point-polygon link is created, from the point added on the receiver to the source (Fig. 14(c)).

5.3 Visibility Updates

Each refinement operation requires an equivalent update in the visibility information contained in the point-polygon links. We again distinguish the two main cases, source refinement and receiver refinement.

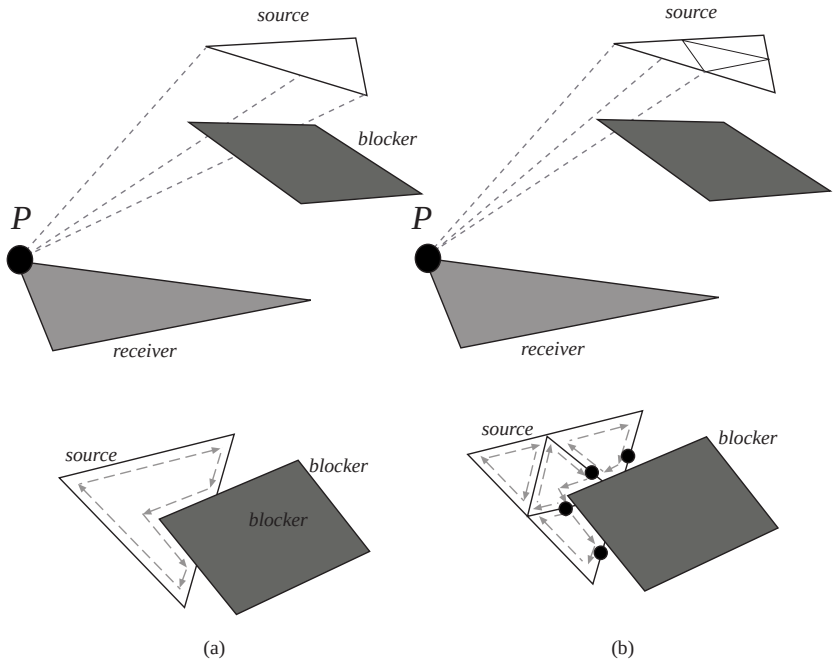


Fig. 15. Source visibility updates. The dashed arrows (lower part) represent the limits of the visible part of the source used to compute the form-factors. (a) The point-polygon link before subdivision and below the corresponding view of the source (b) One of the 4 new point-polygon links due to subdivision and the four new views. The black circles correspond to new nodes of the Skeleton.

In the case of source refinement we need to update the existing visibility information contained in the new point-polygon links. Since the visibility information of such a link can be represented by the view of the source from the receiver point of the link, all that needs to be done is the update of the link with respect to the new source sub-triangles. For example, in Fig. 15(a) the original view from point P is shown in the lower part of the figure. Once the polygon-polygon link is subdivided, four new views are computed, shown in the lower part of Fig. 15(b). The new point-polygon links now contain the references to the skeleton arcs (swaths), corresponding to the parts of the view affected. For example the leftmost source sub-triangle is completely unoccluded from P and thus no arcs are stored. For the others, the intersections of the previously existing arcs and the source sub-triangles result in new skeleton nodes (corresponding to the black circles in Fig. 15(b)). The

corresponding arcs are then subdivided. The new nodes are adjacent to these subdivided arcs. Note that all visibility/view updates are performed in 2D.

Refining a receiver is more involved. When adding a point to a polygon, a new view needs to be computed. We use the algorithm of the Skeleton construction which is robust. The only difference is that we use the blocker lists defined by the arcs stored in the initial point-polygon links instead of the entire model. Since the number of polygons in any given blocker list is relatively small, the cost of computing the new view is low. An example is shown in Fig. 16(a)-(b), where the point P is added to the receiver. In Fig. 16(b) we see the point and the new point-polygon link, and in Fig. 16(c) the newly calculated view is illustrated. The black circles correspond to newly created nodes of the skeleton.

The case of full visibility is detected using the information contained in the polygon-polygon links. The visibility update is then optimized: no new arc is computed and the unoccluded form-factor is used, thus saving time and memory.

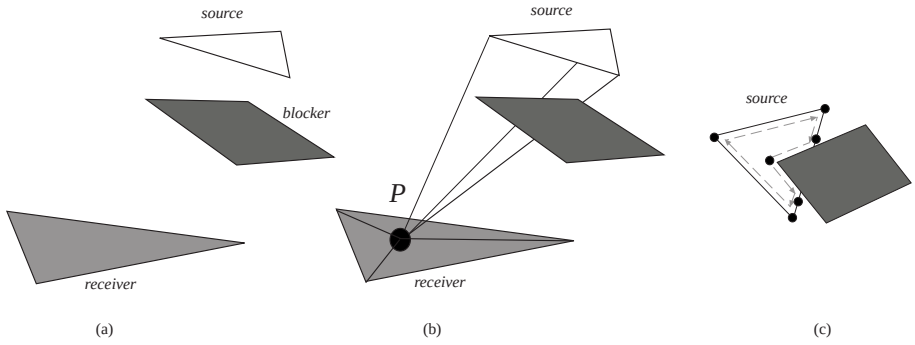


Fig. 16. Receiver visibility updates: (a) The initial configuration. Blocker information is contained in the source-receiver link. (b) A point is added to the receiver, creating a new point-polygon link. (c) The new view of the source computed at P. The blocker lists are updated using this computation.

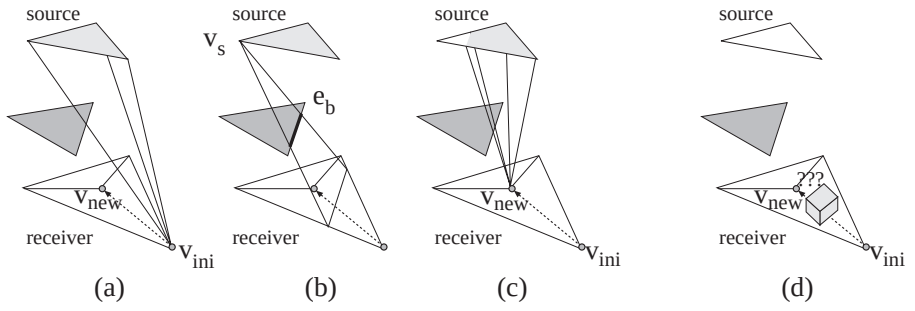


Fig. 17. Receiver refinement with visibility events (this solution was not implemented). (a) We start with a view at one of the initial vertices. (b) We walk across the receiver to the new vertex. Here we cross a ve_b event. v_s begins to be hidden by the blocker. (c) We obtain the view at the new vertex. (d) In the case of touching objects, no information can be kept while crossing the interface between the two objects.

5.4 *Alternative Visibility Updates*

Visibility updates could be performed using the information encoded in the Skeleton, starting from the view at one of the initial vertices, then walking to the new vertex and updating the view each time a visibility event is crossed (see Fig. 17). This method is however hard to implement, and suffers from robustness problems if the visibility events are not crossed in a coherent (if not exact) order. Moreover, the case of touching objects complicates the problem furthermore since no information can be kept while walking "under" a touching object.

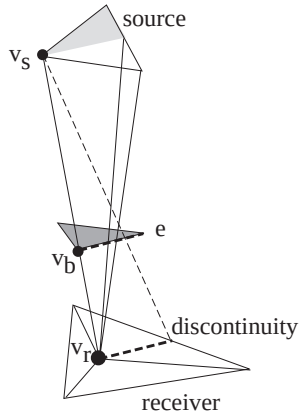


Fig. 18. Degeneracy due to discontinuity meshing. The receiver is split along discontinuity $v_s e$, causing a degenerate $v_s v_b v_r$ extremal stabbing line.

5.5 *Treating Degeneracies*

Subdividing along discontinuities induces degenerate viewpoints. For example in Fig. 18, we subdivide the receiver along the discontinuity $v_s e$. The view from v_r has a degenerate $v_s v_b v_r$ extremal stabbing line. To treat it coherently, we store with each vertex of the triangulation the extremal stabbing line which caused it (which is possibly null). This is a simpler and more robust alternative to the ray-casting modified for grazing objects described in Section 2.2. The treatment of the degeneracy then proceeds in the same manner.

A different alternative would have been to slightly perturb the point position to avoid those degeneracies. Two reasons have prevented us from doing so. First, discontinuity meshing allows us to delimit regions of umbra (full occlusions), regions of full visibility, and regions of penumbra. The two first region types require coarser subdivision than the latter. If we perturb the point position, some regions which should have been totally in the umbra will have a very small part in the penumbra, and need more subdivision. Second, point perturbation would cause numerical precision problems.

6. POLYGON SUBDIVISION

We have now seen how link and visibility information is updated during the light propagation process. Evidently, link updates are a consequence of a *refinement* decision, based on an appropriate criterion. We have chosen to use a *perceptually-based* refinement criterion.

In what follows, we first review basic concepts of perceptual mapping which we use for our refinement criteria. We next present the polygon subdivision process, and then detail the perceptually based refinement criterion which we have used for our algorithm.

6.1 Perceptual Just Noticeable Difference

The work in the field of perception provides us with two important features. First, it permits the conversion of radiometric quantities into displayable colors while preserving the subjective impression a viewer would have when observing the real scene. Second, it allows us to use error thresholds related to the error an observer is able to perceive, which are thus easy to set.

The human eye can deal with a very high dynamic range, while computer displays are usually limited to a 1 to 100cd/m^2 range [Gibson and Hubbold 1997]. The eye adapts itself according to the luminosity of the scene being looked at. This explains why we are able to see dark night scenes as well as very luminous sunny scenes. The tone mapping operation deals with the transformation of high range radiometric quantities into low range display colors, while trying to provide the viewer with the same impression as the real scene. One obvious and simple method is to divide all quantities by the maximum radiosity of the scene. The problem with this approach is that if the light source intensity is halved, the scene will look exactly the same, though we would expect it to seem darker.

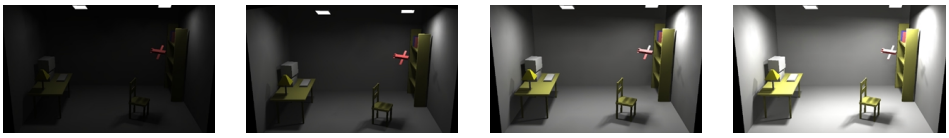


Fig. 19. Effect of Ward's tone-mapping on the same scene with different light source intensities. Note how details remain perceptible while the impression of darkness or luminosity is preserved.

Ward's contrast preserving tone mapping operator [Ward 1994] deals with this problem. A simple scaling factor sf is used for the whole scene which depends on the maximal displayable luminance L_{dmax} and the world adaptation level L_{wa} which is usually the logarithmic average of the scene luminosity without primary light sources. In what follows, all intensities are expressed in $candelas/meter^2$ (a *candela* is a $lumen/steradian$ [Ward 1994]). The scaling factor sf is then given by

$$sf = \frac{1}{L_{dmax}} \left[\frac{1.219 + (L_{dmax}/2)^{0.4}}{1.219 + L_{wa}^{0.4}} \right]^{2.5}$$

.

Fig. 19 demonstrates the effect of this operator on a given scene with different source intensities.

We use a technique similar to that of Gibson *et al.* to compute the adaptation level [Gibson and Hubbold 1997]. We use a static adaptation level which is the average radiosity of the scene. However, since we use hierarchical radiosity as opposed to progressive radiosity, we do not have to rely on an estimate involving the average luminance and reflectance. Instead, at any step we use the average radiosity value of the polygons of the scene. This

is why we start the radiosity computation with several gather steps to compute a coarse estimate of the light distribution.

The use of a global static adaptation level is only a coarse approximation of the human visual system adaptation. As shown by Gibson *et al.* it gives a fairly good estimate of the dynamic adaptation level [Gibson and Hubbold 1997]. More elaborate solutions could be explored, such as the use of local adaptation levels computed using the average radiosity in the neighbourhood of an object, and more involved tone-mapping operators could also be used [Tumblin and Rushmeier 1993; Ferwerda et al. 1996] but this is beyond the scope of this article.

Once the tone mapping operation has been applied, the admissible error can be set as a given percentage of the maximum displayable intensity L_{dmax}. Psychovisual studies [Gibson and Hubbold 1997; Murch 1987] have shown that the human eye is able to distinguish a difference of 2%: this is the *just noticeable difference*. We will call the allowed error ε_{percep}, and in practice we will use $\varepsilon_{percep} = 2\%$ for all our refinement criteria.

```
subdividePolygons() {
   for each polygon r and each poly-poly link s to r
      if shouldRefine Link(s,r)
         refine source
   for each polygon r and each poly-poly link s to r
      if shouldRefine Link(s,r)
         if iteration < 3
            regularSubdivision( r ) // perform grid-like subdivision
         else
            find and insert discontinuities in r
   complete subdivision at this level // create sub-triangles in meshes
}
```

Fig. 20. Polygon Subdivision

6.2 Polygon Subdivision

Our experiments have shown that subdividing along the discontinuities during the first few subdivisions results in the creation of triangles with poor aspect ratios, inducing very visible artifacts. For this reason, subdivision of the polygons is performed using a two step strategy:

—*During the first two subdivisions:* The polygons are subdivided in a regular grid-like manner. In particular, a regular grid is created as a function of size of the polygon being subdivided.

—*During the third and subsequent subdivisions:* Insert shadow discontinuities or other illumination detail. Discontinuities are added as constrained edges, and result in a modified triangulation.

This approach is similar in spirit to the approaches of Stuerzlinger [Sturzlinger 1994] and Hardt and Teller [Hardt and Teller 1996] where the discontinuity meshing is, however, used for display purposes only. The polygon subdivision algorithm is outlined in Figure

20. In contrast to standard hierarchical radiosity, we cannot subdivide the polygons on-the-fly when a link needs subdivision, because polygon subdivision is not uniform and has to be performed along discontinuity curves. For this reason, we first consider all the polygon-polygon links for a given polygon to decide if it requires subdivision, and to determine which discontinuities will be inserted. After all discontinuities for a given receiver have been inserted, the CDT is completed.

6.3 Maxima insertion

If we consider the radiosity of a light source as a function defined over a receiver, it has been shown that subdividing a mesh used to represent illumination at the maximum of the function can increase the accuracy of the radiosity solution [Drettakis and Fiume 1993]. We thus first compute the maxima of the unoccluded radiosity functions of the light sources before the first refinement.

The maxima are computed only for important light-transfers (estimated using the disk-disk formula [Hanrahan et al. 1991] and the perceptual metric). Given a receiver and a polygon considered as a source, we use a gradient-descent algorithm to locate the maximum. Once the maximum is found, we compute the contribution of the source at this point; if it is above ε_{percep} the maximum is stored to be subsequently inserted in the mesh. The radiosity of the receiver polygon is updated to take this maximum into account. That is, a gather is performed at the maximum (before a link is created from it) to obtain a better estimate of the light distribution that will be used for the first refinement.

The maxima are inserted as a separate initial step during the first subdivision. The points of the regular subdivision which are too close to a maximum are not inserted. An example was shown in Fig. 4(b), where the maximum corresponds to the point on the lower left which is not exactly on the grid. We thus obtain nearly regular meshes with well shaped triangles.

The maxima-search process is applied iteratively to take indirect illumination into account. The insertion of maxima of indirect sources is very important for example in Fig. 23, where the table (illuminated by the lamp) is the most important light source for the upper part of the left wall.

6.4 Refinement Criterion

We distinguish two refinement criteria (or *oracles*): a radiometric criterion which accounts for the variation of the unoccluded radiosity, and a visibility (discontinuity) criterion. Moreover, the discontinuity criterion also guides the choice of the discontinuity curves to be inserted.

The radiometric oracle estimates if the linear interpolation of the light transfer is "accurate enough". We sample the unoccluded form-factor (see [Baum et al. 1989] and Section 4.2.1) at the center of the patch and at the edge mid-points, and compare this to the linearly interpolated value. If the perceptually transformed difference is larger than ε_{percep}, we proceed with subdivision.

The principle of our visibility oracle is to estimate (as a percentage of the maximum displayable intensity) the "shadow amount" cast by the blockers, that is the radiosity that would be transfered without the blocker. Our refinement criterion thus has three steps: unoccluded estimate, "shadow amount" estimate and "shadow sharpness" estimate.

Consider a receiver and a source. Recall that a source is any polygon in the scene, considered as a source at this step of the refinement process. In what follows we refer to

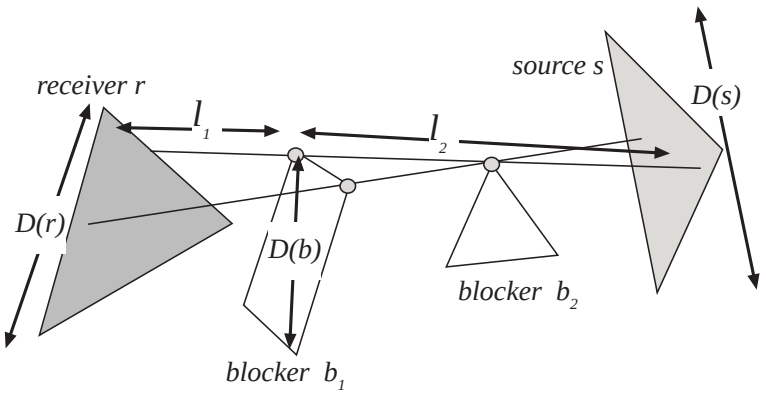


Fig. 21. Refinement criterion geometry

Fig. 21, for the definition of all geometric quantities.

First, we compute an estimate of the unoccluded light transfer B_{unoc} using the disk-disk formula [Hanrahan et al. 1991]. As above, if the estimate is less than ε_{percep}, the link will not be subdivided.

Second, we consider each visibility event between the source and the receiver, and estimate the "shadow amount". To do this, we estimate the part of the source potentially hidden by the blocker by using the projected diameter of the blocker on the source to estimate its projected umbra:

$$D_{proj}(b) = D(b) * \frac{l_2}{l_1}$$

.

The estimated percentage of occlusion is then:

$$occlu = \frac{\frac{\pi}{4} D_{proj}(b)^2}{Area_{source}}$$

,

(clamped to 1). The "shadow amount" is:

$$shadow = B_{unoc} * occlu$$

.

If *shadow* is below ε_{percep}, the visibility event is ignored.

Third, we estimate the sharpness of the shadow. The extent of the zone of penumbra is approximated by projecting the diameter of the source onto the receiver:

$$D(penumbra) = D(s) * \frac{l_1}{l_2}$$

.

If the size of the receiver is bigger than the zone of penumbra, then the receiver may contain regions where the source is completely visible, and regions where the blocker

projects entirely on the source. In the latter case, the fraction of occlusion is maximal and approximated by *occlu*. The variation of radiosity on the receiver is thus equal to *shadow*. Otherwise, we make the approximation that the radiosity varies linearly in the penumbra, and the variation of radiosity on the receiver is then:

$$\Delta(B) = \begin{cases} shadow & if \;\; D(penumbra) > D(r) \\ shadow * D(r)/D(penumbra) & otherwise \end{cases}$$

All the links containing visibility events with $\Delta(B) > \varepsilon_{percep}$ will be subdivided. As explained above, in the first two iterations subdivision will be regular. During the third and later iterations, subdivision is performed by inserting the discontinuities with the highest $\Delta(B)$. This is the *ranking* phase of our algorithm, similar in spirit to that of [Hardt and Teller 1996].

$\Delta(B)$ is computed using l_1 and l_2 at the two extremities of the visibility event, and taking the maximum. Note that the evaluation of these oracles is very rapid since the links and events are pruned as soon as we can decide that they will not cause subdivision.

7. IMPLEMENTATION AND RESULTS

7.1 Implementation

We have used the C++ Visibility Skeleton implementation of [Durand et al. 1997], with the extensions and changes described in Section 2. The scene polyhedra are represented using a winged-edge data-structure and a pre-processing step is performed to detect touching objects, which is a necessary step for the treatment of degeneracies.

The hierarchical triangulation has been incorporated into the same system. We use the public domain implementation of [Guibas and Stolfi 1985] by Dani Lischinski [Lischinski 1994] for the constrained Delaunay triangulation. Each subdivided polygon contains a triangulation `QuadMesh`.

On our test scenes, the algorithm spends most of its time on the visibility update, especially the calculation of the views at new vertices for the receiver refinement. For example, for the Desk scene of Fig. 22, for the last iteration, the computation of the criterion and the refinement of the mesh took 15 seconds, updating the visibility took 64 seconds, and the gather/push-pull took 2.5 seconds.

We have chosen not to use textures in our examples since they usually hide the accuracy of the lighting simulation.

7.2 Results

We present results for four different scenes. The first scene is a simple “Desk” scene, containing 438 polygons and two large, powerful light sources (see Fig. 22). This scene is used to illustrate the general functionality of our algorithm. The second scene contains the same geometry, but with 8 additional small, powerful light sources. The two large light sources have been turned down in intensity; we call this scene “Many Lights” (see Fig. 23). This scene shows how our approach treats the case of multiple light sources effectively. The third scene has been chosen to demonstrate the performance of our algorithm for mainly indirect lighting. We chose a common example of a bedroom lit exclusively by a small downwards-pointing, bed-side lamp. Most of the room is lit indirectly; this scene is called “Indirect” (see Fig. 25). Finally, simply in the interest of showing a completely different type of scene, we show the result of our approach on a “Village” scene, containing

buildings and cars. The scene is lit overhead by a rectangular light (see Fig. 28). In what follows we present various performance statistics as well as an informal comparison with hierarchical radiosity using quadtree subdivision, but with improved refinement and error bound strategies ([Gibson and Hubbold 1996; Lischinski et al. 1994]).

All times presented are in seconds on an R10000 195 MHz Silicon Graphics Onyx 2 workstation.

Before presenting the results for the complete algorithm, we present some interesting statistics concerning the importance of accurate visibility for form-factor computation.

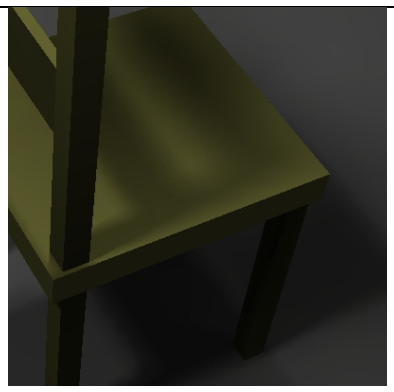
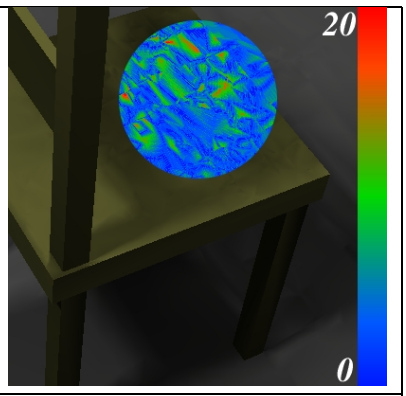
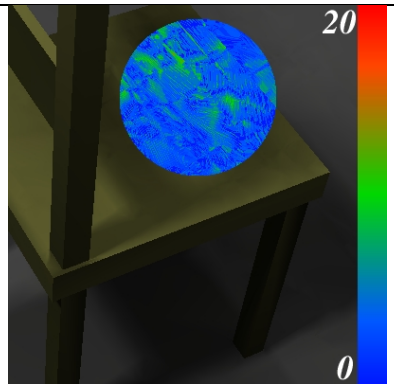
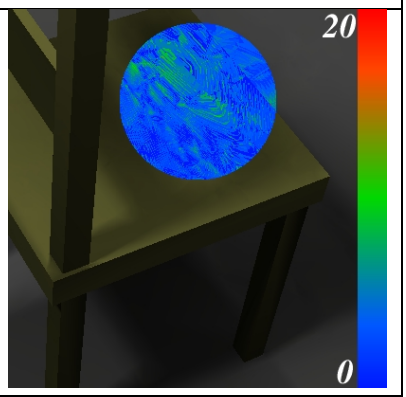
Image		
Method	exact (Skeleton)	16 rays
Total time	1min 19	1min 17
Image		
Method	36 rays	64 rays
Total time	2min 02	3min 04

Table 1. Importance of the form-factor accuracy on a small scene of 246 polygons. The number of rays for the indirect illumination is set to 4, while only the number used for direct illumination varies. In inset we show in false color the difference with the skeleton solution in the perceptually uniform *CIE L*a*b** color space.

7.2.1 *Importance of accurate visibility.* We have run some tests with approximate visibility to judge the importance of the exact computation of the form-factors on the quality of the images. We have slightly modified our implementation to compute the form-factors using ray-casting on a jittered grid sampling of the source. In Table 1, the same method

is used for discontinuity-based mesh subdivision for all cases shown. It is performed using the Skeleton, and the cost of the skeleton construction and update is not included in the ray-casting timings, which report exclusively the cost of form-factor computation. As mentioned before (Fig. 10), inexact form-factor computation introduces significant error. The visual consequences of this can be seen in Table 1. Moreover, the computation overhead is significant if high quality is required because at least 64 rays are needed per form-factor.

Note that the effect on the images is particularly dramatic, because the subdivision induced by the discontinuity meshing is not uniform. The thin triangles introduce very visible artifacts. These results confirm those observed in [Drettakis and Sillion 1996].

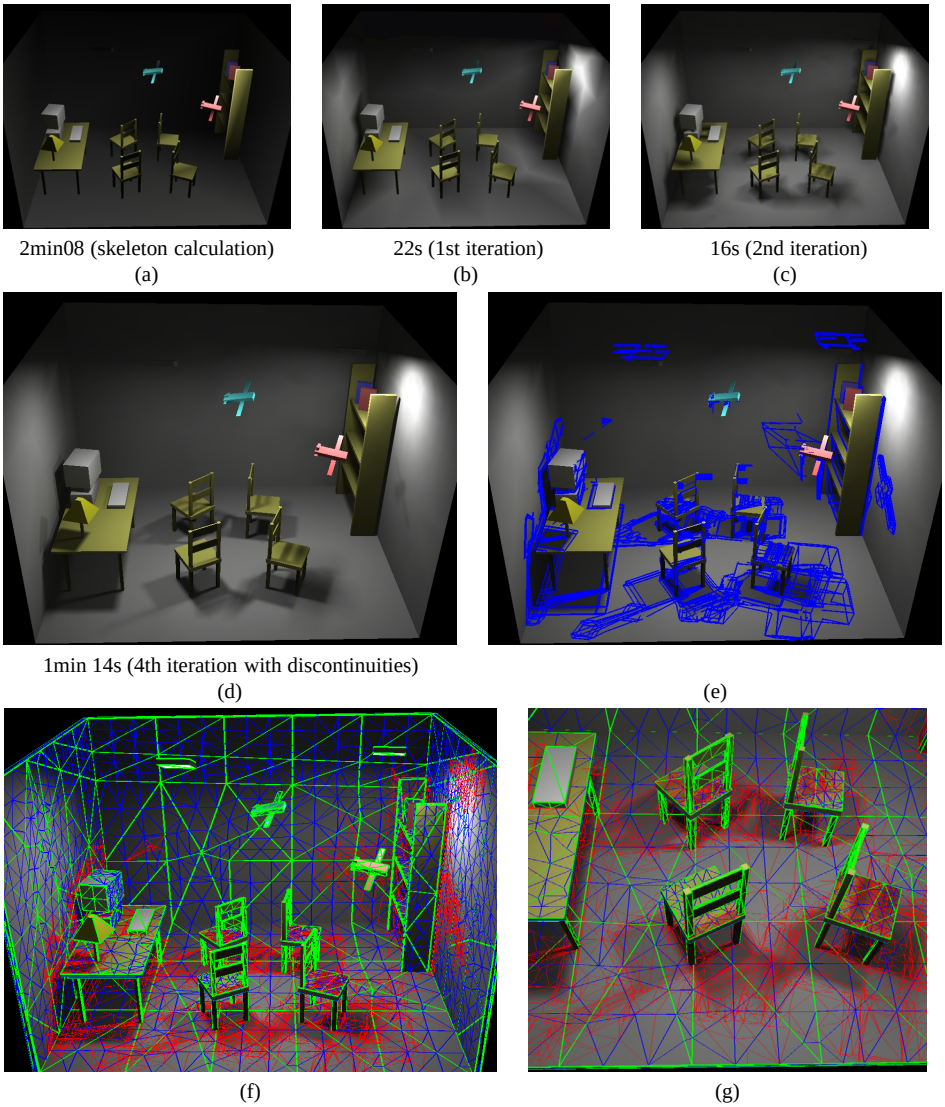
Scene	*Pol*	*Skel*	*1st*	*2nd*	*3rd*	*Total*	*Mem(ini/tot)*	*links*	tris
Desk	444	2min 08	22s	16s	1min 14	4min	40/200MB	378K	46K
Many	492	2min 23	2min 38	55s	4min 27	10min 23	47/365MB	1546K	104K
Bed	534	4min 12	1min 25	58s	4min 35	11min 10	56/400MB	383K	43K
Village	312	45s	12s	7s	24s	1min 28	15/43MB	134K	28K

Table 2. Timing and memory results for the test scenes. The memory statistics shown are the initial memory usage for the skeleton *before* any subdivision, and the total memory used after the subdivision for lighting.

7.2.2 *General Solution.* The images of Fig. 22 show the initial steps of the algorithm as described previously. Fig. 22(a) is the result of three gather steps on the initial unsubdivided scene. Note that at this point we already have a very crude approximation of the global distribution of illumination in the scene, since the form-factors at the vertices are exact. In Fig. 22(b) we see the first step which is a regular grid together with the maxima of the light sources inserted into the mesh. Fig. 22(c) and (d) show the evolution of the algorithm after two iterations. The shaded images without the meshes are shown in (d). In (e) we show the discontinuities actually inserted. Note that these include discontinuities for all light transfers (direct and indirect) and that their number is much lower than that for a discontinuity meshing type approach (about 40% of the discontinuities caused by direct sources have been inserted).

In Table 2 we show the statistics of scenes computed using our method. For the “Desk” scene, we see that the total solution, including illumination, requires 4 minutes of computation. The quality of the solution is very high, including well-defined shadows on all surfaces. Note high quality shadows on the chairs and the table. The total number of point-polygon links is 378,746, and the number of leaf triangles is 46,058.

7.2.3 *Treating Many Lights.* One scene type for which our approach performs particularly well is that of multiple sources. This is demonstrated by our second test scene containing 10 lights and the same geometry as “Desk”. Fig. 23(a) shows an overview of the scene as rendered by our new approach, and Fig. 23(c) shows a closeup of the floor. The shadows due to the multiple sources are well represented in the areas when appropriate. The perceptually based ranking algorithm has correctly chosen the discontinuities that are of importance, since the combined influence of all sources is taken into account. This is shown by the small number of discontinuities present on the floor in Fig. 23(d). From Table 2 we see that 1.5 million links were used in this scene and the total computation time was 10 minutes 23 seconds. Only 10% of the direct discontinuity segments have been inserted.



2min08 (skeleton calculation) (a)
22s (1st iteration) (b)
16s (2nd iteration) (c)
1min 14s (4th iteration with discontinuities) (d)
(e)
(f)
(g)

Fig. 22. Initial Desk Scene. In (a) we show the initial, unsubdivided scene. In (b) we show the first step which includes the grid and the maxima, in (c) we show the second iteration and (d) show the results of the third iteration which includes the discontinuity meshing. (e) shows the discontinuities actually inserted. (f) and (g) show the hierarchical triangular mesh (first level in green, second in blue, and third in red).

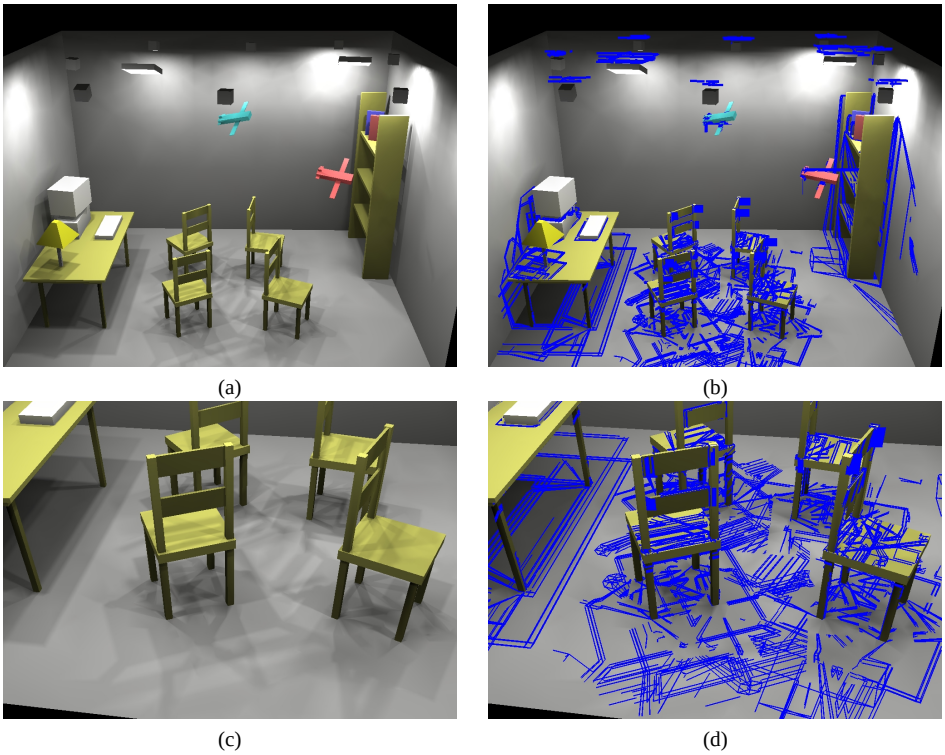
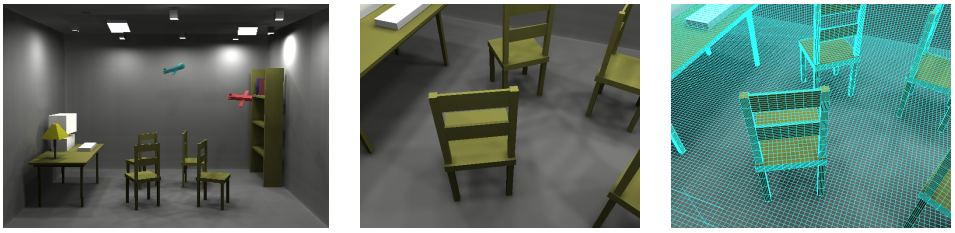


Fig. 23. Many Lights scene: (a) the final image, (b) the discontinuities actually inserted. (c) and (d) a closeup view of the floor.

As an informal comparison, we have compared to an implementation of hierarchical radiosity with clustering with the refinement proposed by Gibson and Hubbold [Gibson and Hubbold 1996] using the error bound propagation of Lischinski *et al.* [Lischinski et al. 1994]. For the Many lights scene computation with 1 million links, the computation time is almost 2 hours (Table 3). In addition, the quality of the results is lower, since the multiple shadows are much less sharp, or even missing (see Fig. 24). A much larger number of links would be necessary to compute an image of similar quality to Fig. 23 using hierarchical radiosity. Note that despite the fact that this method uses approximately the same number of elements (110K *vs.* 104K for our method), the quality of the resulting images is much lower.

Scene	*Pol*	*1st*	*2nd*	*3rd*	4th	*Total*	*Mem*	*links*	elems
Many	492	1 hr 25	22 min	10 min	-	1 hr 57	147 Mb	1098K	110K
Bed	534	11 min	37 min	6 min	25s.	54 min	94 Mb	903K	32K

Table 3. Comparative Timing and memory results for the test scenes using Hierarchical Radiosity with error bounds [Lischinski et al. 1994] and Gibson and Hubbold [Gibson and Hubbold 1996] refinement.



(a) (b) (c)

Fig. 24. Hierarchical Radiosity comparative results for Many Lights scene: (a) the final mesh, (b) a general view of the rendered scene (c) a closeup view of the floor.

7.2.4 *Indirect Illumination.* Accurate and efficient computation for indirect lighting is another challenge for our approach. It is for this type of scene that we see the power of our accurate form-factor and discontinuity ranking method. Previous approaches require significantly longer computation time to achieve this level of precision for secondary illumination.

This is illustrated with our third test scene (Fig. 25 and 26), in which light arrives from the bedside lamp which is pointing downwards only (no light leaves from the sides or the top of the lamp). Thus everything in the room above the level of the lamp is lit indirectly.

The algorithm uses a relatively small number of point-polygon links (383,715), and manages to represent shadows generated by secondary illumination. Notice for example the shadows of the right hand lamp or the books on the far wall in Fig. 25(d); these are caused by illumination of light bouncing off the bedside table and the bed.

Another informal comparison is presented, using the same algorithm as described above (based on [Gibson and Hubbold 1996; Lischinski et al. 1994]). Using almost a million links, hierarchical radiosity takes a slightly less than one hour, and produces lower quality results (see Fig. 27(a) and (b)).

Moreover, the advantages of our linear lazy-wavelet representation are well illustrated on the overall view of the hierarchical radiosity solution. The left part of the back wall is much lighter than the right part, with a strong discontinuity inbetween revealing the quadtree nature of the mesh. This is because interpolation is applied as a post-process at the finest level of subdivision; exchanges simulated at higher level are thus not correctly interpolated.

7.2.5 *Village Scene.* A final scene of a village is shown in Fig. 28, to show that the algorithm can be used for different scene types. Here the scene is lit overhead by a rectangle and also by the head and rear lights of the cars.

8. DISCUSSION

Our new approach shows promising results in what concerns the representation of accurate shadows, in particular for the cases of multiple sources and indirect lighting. However, the method presented is not without limitations. We believe that it is worthwhile to review what we consider to be the most important limitations and drawbacks as well as the most important advantages and contributions of our approach.

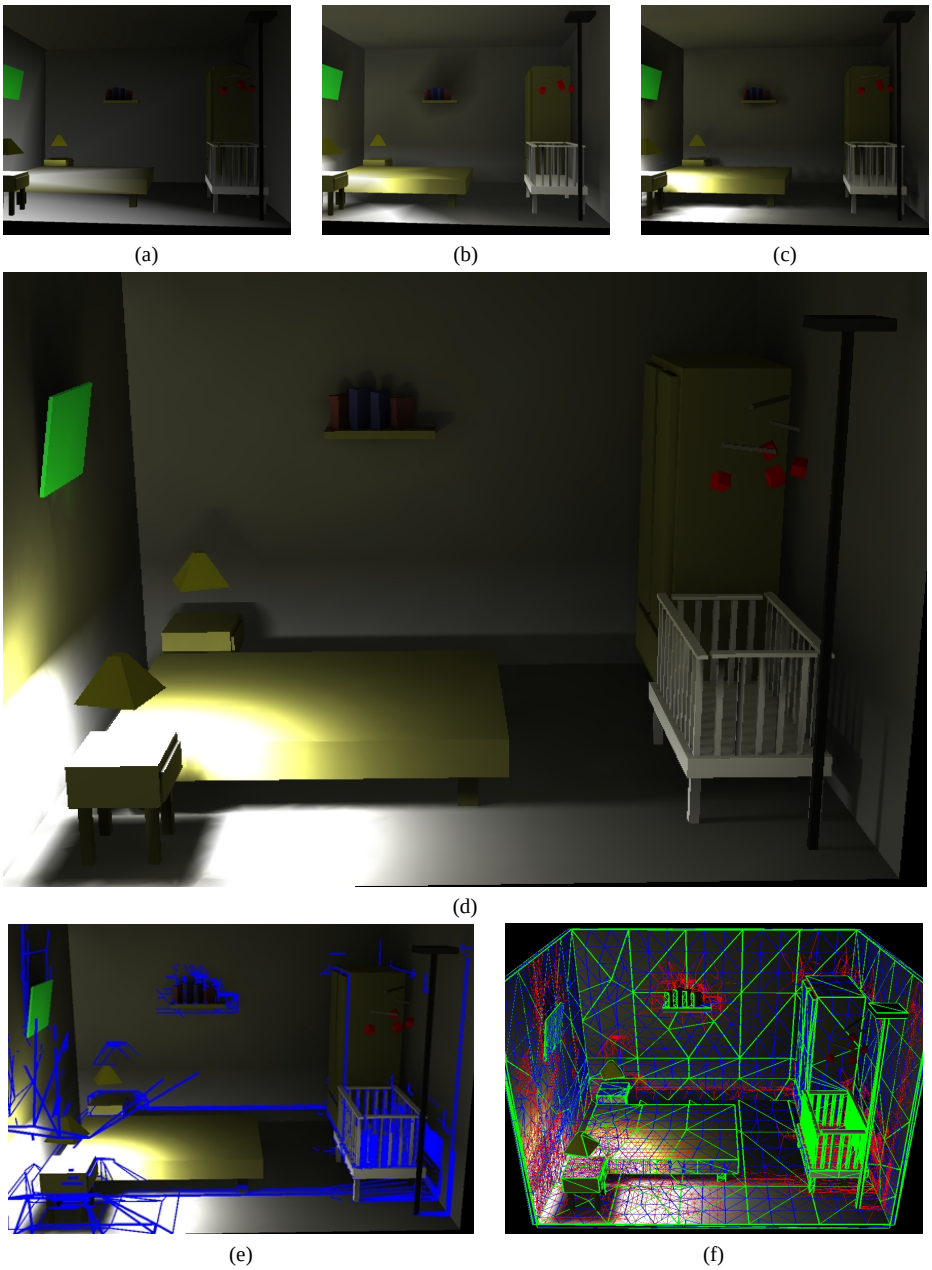


Fig. 25. Indirect lighting scene: (a) Initial solution, (b) first iteration (c) second iteration (d) final image (e) discontinuities inserted (the discontinuities inserted on the front wall are represented though this wall is backface-culled) (f) hierarchical triangulation.

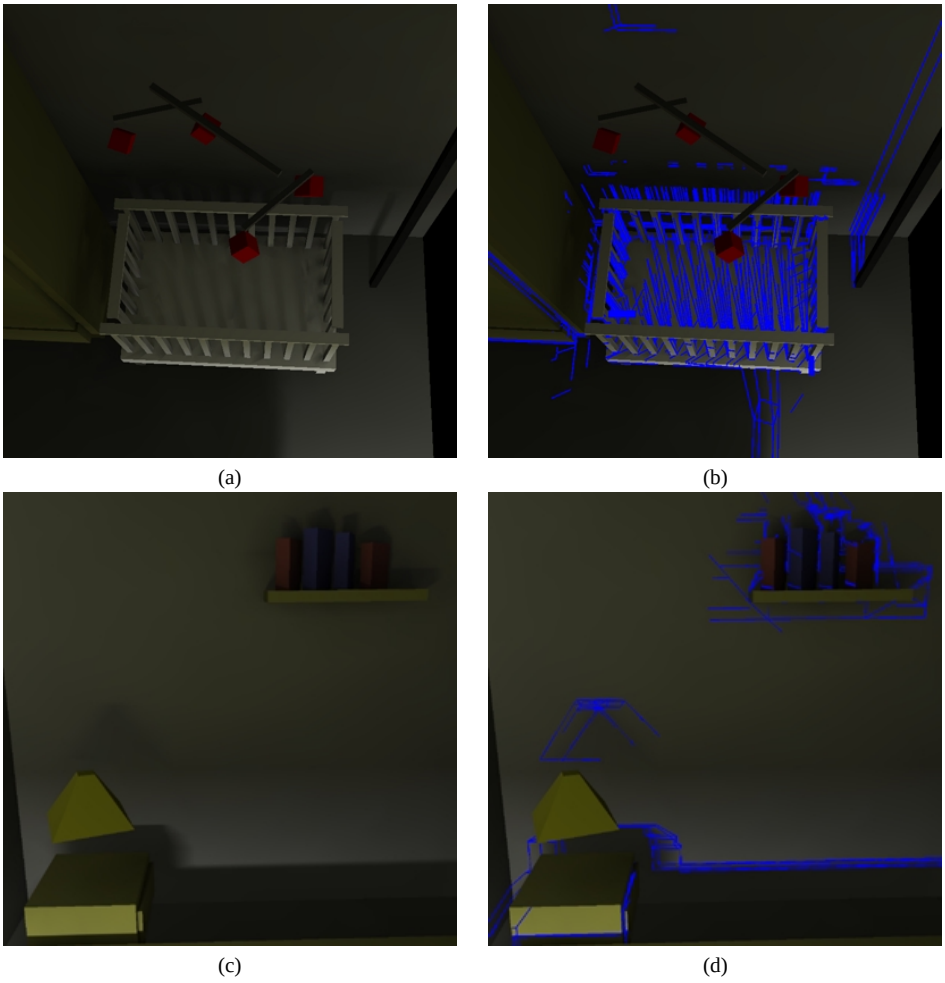


Fig. 26. Indirect lighting scene: (a) and (b) closeup of the right wall (c) and (d) closeup of the back wall. The lower part of the wall is directly illuminated by the left lamp (which is not visible on this image), while the upper part is indirectly illuminated by the left table. Note the indirect shadows cast by the books and the right lamp.

8.1 Limitations

Two major limitations of this work can be identified, the first is high memory consumption and the second is numerical robustness problems of the algorithms used.

The memory usage of the skeleton data structure is high, and can often have quadratic growth in the number of input polygons, depending on the how complex the visibility relations are between polygons. Even for simple environments, our method uses very large amounts of memory (see Table 2). To make our approach practical for large scenes, it is evident that we need to adopt one or a combination of the following strategies: lazy or on-demand skeleton construction, divide-and-conquer strategies (similar to *e.g.*, [Hardt and Teller 1996]) or a clustering approach allowing a multi-resolution representation. Some

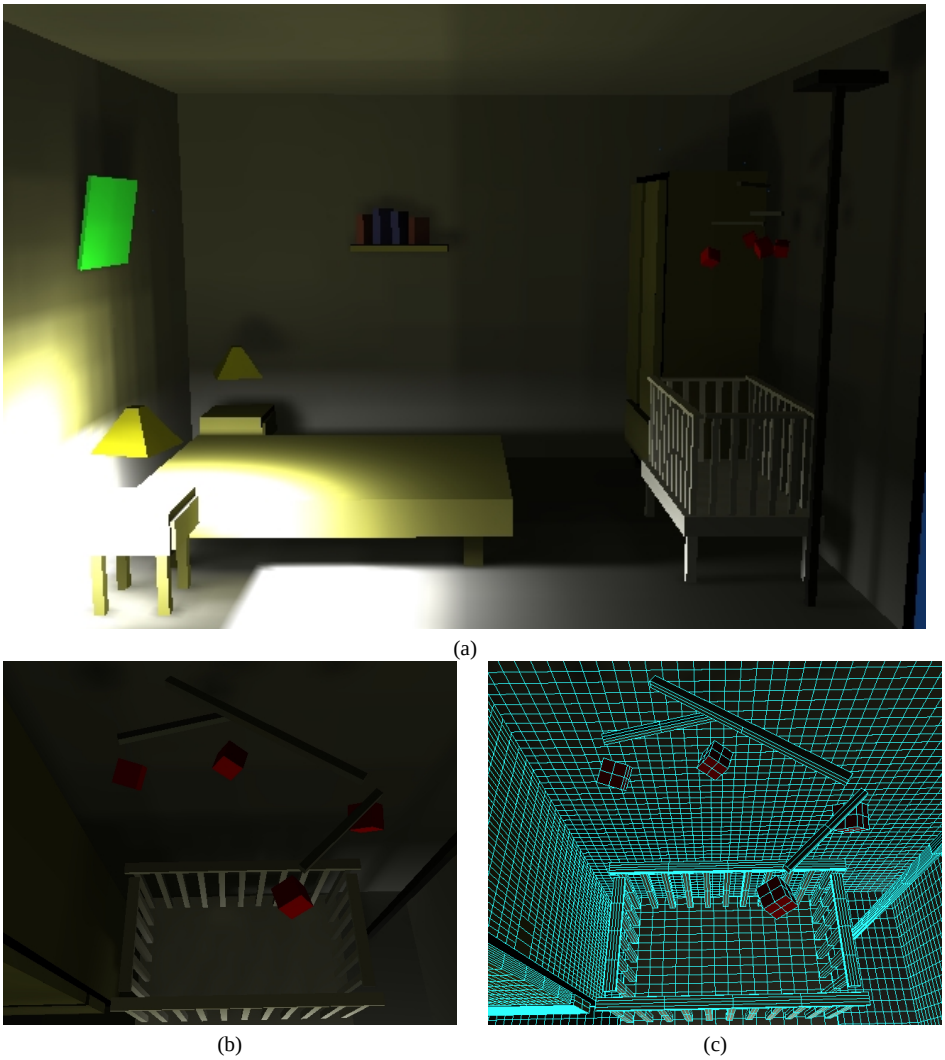
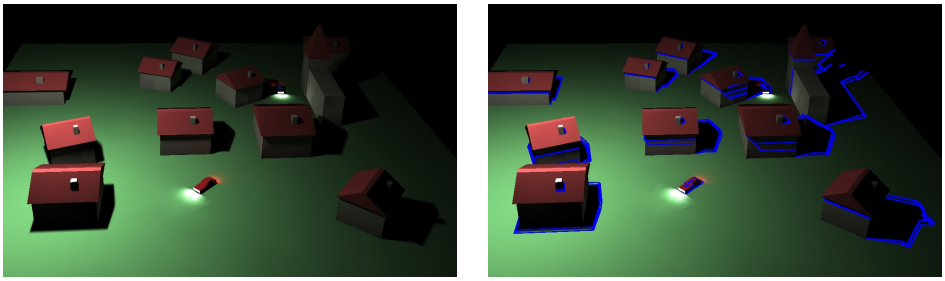


Fig. 27. Hierarchical Radiosity comparative results for Indirect Lighting scene: (a) the final image, (b) and (c) a closeup view of the right-hand wall.

ideas in these direction can be found in the Section 9 on future work.

Numerical robustness and the treatment of degenerate cases are important issues. Despite the simplicity of the construction algorithm which is based on ray-casting for node determination, degenerate cases can cause problems. As discussed in Section 2.2 we have been able to reliably treat most of these. Nonetheless, in the case of subdivision, many visual events coincide, causing problems of coherence both for the ray-tracing step (for node creation) and the adjacency determination. These problems are particularly evident in the case of view updates. A coherent and consistent treatment of degeneracies is planned, but is a research topic in itself and beyond the scope of this paper (see Section 9). Insertion



(a) (b)

Fig. 28. Village scene (a) final image (b) discontinuities actually inserted

of points in the mesh also causes problems, especially during subdivision due to numerical imprecision. Symbolic calculations instead of numerical intersections could potentially resolve most of these problems.

8.2 Advantages

The visibility-driven hierarchical radiosity algorithm introduced here has many advantages. First we achieve visually accurate shadows using discontinuities and exact point-to-polygon form-factors, for both direct and indirect illumination. The new hierarchy of triangulations data-structure, the novel two link types and the multi-resolution point-area link representation allow accurate linear reconstruction of radiosity over irregular meshes. The global treatment of visibility and discontinuities permits the definition of an efficient refinement oracle. Using a perceptually based method to estimate shadow importance, our refinement algorithm has proven to be very efficient for previously hard-to-handle scenes such as scenes lit with multiple light sources and scenes lit mainly by indirect light. As part of an informal comparison, we have seen that Hierarchical Radiosity uses more computation time to produce much lower quality results, as would be expected.

Approaches such as that of [Lischinski et al. 1993] based on discontinuity meshing have difficulty with large numbers of light sources, since the number of discontinuities becomes unmanageable very quickly. This has consequences both on computation time and on robustness in the construction of the discontinuity mesh. For similar reasons, no discontinuity-based *hierarchical* lighting algorithm has been proposed previously in which discontinuities are treated for indirect light transfers.

9. CONCLUSION AND FUTURE WORK

We have presented a new hierarchical radiosity algorithm using the extended Visibility Skeleton. We have extended the Skeleton by replacing the n^2 table representation of the nodes and arcs by a structure of hierarchical links from polygons to polygons (and vertices to polygons). We have introduced update algorithms permitting the maintenance of consistent views at vertices added to a polygon due to subdivision, as well as the resulting sub-faces.

These extensions result in a powerful data structure which permits the computation of exact point-to-polygon form-factors for any vertex/polygon pair in the scene, and which provides detailed visibility information between any (sub)polygon-(sub)polygon pair.

We have introduced a novel hierarchical radiosity algorithm using this structure, based on a "lazy wavelet" or "sub-sampling" type multi-resolution representation. The basic data structure used is a non-uniform hierarchical triangulation, which consists of a hierarchy of embedded constrained Delaunay triangulations. By maintaining radiosity differences at subdivided vertices, we introduce a linear "push" step, resulting in higher quality radiosity reconstruction at the leaves. A new, perceptually-based, discontinuity driven refinement criterion has also been introduced, resulting in hierarchical subdivision of surfaces well adapted to shadow variations. The results of our implementation show that we can generate accurate high-quality, view-independent solutions efficiently. The results also show that our approach is particularly well suited to previously hard-to-handle cases such as multiple light sources and scenes lit almost entirely by indirect illumination.

Future Work

As was the case with the initial Skeleton work [Durand et al. 1997], memory usage remains the major limitation of the visibility skeleton. It is clear that a clustering-type approach is required, which will allow us to apply our algorithm to the parts of the scene where it is required. The idea would be to compute a visibility skeleton inside each cluster and approximate visibility skeletons between clusters. The challenge is to define this approximate skeleton, since clusters are not opaque objects.

Moreover, we believe that this clustering approach is a promising way of solving the robustness problem. If the objects are grouped into clusters of a given size, it is easier to set an epsilon for the computations inside this cluster and decide which error is acceptable. In addition, since the number of objects would be almost constant inside clusters, specific verification algorithms could be applied.

The advantage of the skeleton construction is that it is local, and thus can be built in a "lazy" or even "on-demand" fashion. Using to-be-defined criteria, we could compute only the parts of the visibility skeleton related to "important" light transfers. This information could be deleted once used, thus dramatically reducing the memory requirements.

The skeleton could also be used for Monte-Carlo methods. In the case of standard Monte-Carlo techniques, the inherent random nature of the sampling makes it hard to take coherence into account. However, more recent approaches such as quasi Monte-Carlo radiosity [Keller 1996], photon maps [Jensen 1996] or Metropolis light transport [Veach and Guibas 1997] could be coupled with the skeleton for a better exploration of the path space.

Extending the skeleton and the resulting illumination algorithm to dynamic scenes is another promising research direction. The notion of visual events can be extended to temporal visual events, for example when one line goes through five edges of the scene.

ACKNOWLEDGMENTS

We wish to thank Seth Teller for the discussions we had about visibility, Dani Lischinski for his insights on radiosity and discontinuity meshing, and François Sillion and Cyril Soler for all their answers about hierarchical radiosity. This research was funded in part by the European Union ARCADE Reactive LTR project #24944.

References

BAUM, D. R., RUSHMEIER, H. E., AND WINGET, J. M. 1989. Improving radiosity solutions through the use of analytically determined form-factors. In J. LANE

Ed., *Computer Graphics (SIGGRAPH '89 Proceedings)*, Volume 23 (July 1989), pp. 325–334.

BEKAERT, P., NEUMANN, L., NEUMANN, A., SBERT, M., AND WILLEMS, Y. 1998. Hierarchical monte carlo radiosity. In G. DRETTAKIS AND N. MAX Eds., *Eurographics Rendering Workshop 1998* (New York City, NY, June 1998). Eurographics: Springer Wein.

BOUATOUCH, K. AND PATTANAIK, S. N. 1995. Discontinuity Meshing and Hierarchical Multiwavelet Radiosity. In W. A. DAVIS AND P. PRUSINKIEWICZ Eds., *Proceedings of Graphics Interface '95* (San Francisco, CA, May 1995), pp. 109–115. Morgan Kaufmann.

CHRISTENSEN, P. H., STOLLNITZ, E. J., SALESIN, D. H., AND DEROSE, T. D. 1996. Global illumination of glossy environments using wavelets and importance. *ACM Transactions on Graphics 15*, 1 (Jan.), 37–71. ISSN 0730-0301.

DE FLORIANI, L. AND PUPPO, E. 1995. Hierarchical triangulation for multiresolution surface description geometric design. *ACM Transactions on Graphics 14*, 4 (Oct.), 363–411. ISSN 0730-0301.

DRETTAKIS, G. AND FIUME, E. 1993. Accurate and consistent reconstruction of illumination functions using structured sampling. *Computer Graphics Forum (Eurographics '93) 13*, 3 (Sept.), 273–284.

DRETTAKIS, G. AND FIUME, E. 1994. A fast shadow algorithm for area light sources using backprojection. In A. GLASSNER Ed., *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series (July 1994), pp. 223–230. ACM SIGGRAPH: ACM Press. ISBN 0-89791-667-0.

DRETTAKIS, G. AND SILLION, F. 1996. Accurate visibility and meshing calculations for hierarchical radiosity. In X. PUEYO AND P. SCHRÖDER Eds., *Eurographics Rendering Workshop 1996* (New York City, NY, June 1996), pp. 269–278. Eurographics: Springer Wein. ISBN 3-211-82883-4.

DURAND, F., DRETTAKIS, G., AND PUECH, C. 1996. The 3D visibility complex: A new approach to the problems of accurate visibility. In X. PUEYO AND P. SCHRÖDER Eds., *Eurographics Rendering Workshop 1996* (New York City, NY, June 1996), pp. 245–256. Eurographics: Springer Wein. ISBN 3-211-82883-4.

DURAND, F., DRETTAKIS, G., AND PUECH, C. 1997. The visibility skeleton: A powerful and efficient multi-purpose global visibility tool. In T. WHITTED Ed., *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series (Aug. 1997), pp. 89–100. ACM SIGGRAPH: Addison Wesley. ISBN 0-89791-896-7.

FERWERDA, J. A., PATTANAIK, S., SHIRLEY, P., AND GREENBERG, D. P. 1996. A model of visual adaptation for realistic image synthesis. In H. RUSHMEIER Ed., *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series (Aug. 1996), pp. 249–258. ACM SIGGRAPH: Addison Wesley. held in New Orleans, Louisiana, 04-09 August 1996.

GIBSON, S. AND HUBBOLD, R. J. 1996. Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum 15*, 5, 297–310. ISSN 0167-7055.

GIBSON, S. AND HUBBOLD, R. J. 1997. Perceptually-driven radiosity. *Computer Graphics Forum 16*, 2, 129–141. ISSN 0167-7055.

GIGUS, Z., CANNY, J., AND SEIDEL, R. 1991. Efficiently computing and repre-

senting aspect graphs of polyhedral objects. *IEEE PAMI 13*, 6, 542–551.

GORTLER, S. J., SCHRODER, P., COHEN, M. F., AND HANRAHAN, P. 1993. Wavelet radiosity. In *Computer Graphics Proceedings, Annual Conference Series, 1993* (1993), pp. 221–230.

GUIBAS, L. AND STOLFI, J. 1985. Primitives for the manipulation of general subdivisions and computation of voronoi diagrams. *ACM Transactions on Graphics 4*, 2 (April), 74–123.

HANRAHAN, P., SALZMAN, D., AND AUPPERLE, L. 1991. A rapid hierarchical radiosity algorithm. In T. W. SEDERBERG Ed., *Computer Graphics (SIGGRAPH '91 Proceedings)*, Volume 25 (July 1991), pp. 197–206.

HARDT, S. AND TELLER, S. 1996. High-fidelity radiosity rendering at interactive rates. In X. PUEYO AND P. SCHRÖDER Eds., *Eurographics Rendering Workshop 1996* (New York City, NY, June 1996), pp. 71–80. Eurographics: Springer Wein. ISBN 3-211-82883-4.

HECKBERT, P. 1992. Discontinuity meshing for radiosity. *Third Eurographics Workshop on Rendering*, 203–226.

HEDLEY, D., WORRALL, A., AND PADDON, D. 1997. Selective culling of discontinuity lines. In J. DORSEY AND P. SLUSALLEK Eds., *Rendering Techniques '97* (8th EG workshop on Rendering, Saint Etienne, France, June 1997), pp. 69–81. Springer Verlag.

JENSEN, H. W. 1996. Global illumination using photon maps. In X. PUEYO AND P. SCHRÖDER Eds., *Eurographics Rendering Workshop 1996* (New York City, NY, June 1996), pp. 21–30. Eurographics: Springer Wein. ISBN 3-211-82883-4.

KELLER, A. 1996. Quasi-monte carlo radiosity. In X. PUEYO AND P. SCHR ÖDER Eds., *Eurographics Rendering Workshop 1996* (New York City, NY, June 1996), pp. 101–110. Eurographics: Springer Wein. ISBN 3-211-82883-4.

LISCHINSKI, D. 1994. Incremental delaunay triangulation. In P. S. HECKBERT Ed., *Graphics Gems IV*, pp. 47–59. San Diego, CA: Academic Press Professional.

LISCHINSKI, D., SMITS, B., AND GREENBERG, D. P. 1994. Bounds and error estimates for radiosity. In A. GLASSNER Ed., *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series (July 1994), pp. 67–74. ACM SIGGRAPH: ACM Press. ISBN 0-89791-667-0.

LISCHINSKI, D., TAMPIERI, F., AND GREENBERG, D. P. 1992. Discontinuity meshing for accurate radiosity. *IEEE Computer Graphics and Applications 12*, 6 (Nov.), 25–39.

LISCHINSKI, D., TAMPIERI, F., AND GREENBERG, D. P. 1993. Combining hierarchical radiosity and discontinuity meshing. In *Computer Graphics Proceedings, Annual Conference Series, 1993* (1993), pp. 199–208.

MARTIN, I., PUEYO, X., AND TOST, D. 1997. An Image Space Refinement Criterion for Linear Hierarchical Radiosity. In *Proceedings of Graphics Interface '97* (San Francisco, CA, May 1997). Morgan Kaufmann.

MITCHELL, D. P. 1996. Consequences of stratified sampling in graphics. In H. RUSHMEIER Ed., *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series (Aug. 1996), pp. 277–280. ACM SIGGRAPH: Addison Wesley. held in New Orleans, Louisiana, 04-09 August 1996.

MURCH, G. 1987. olor displays and color science. In H. J. DURRET Ed., *Color and the Computer*, pp. 1–25. Boston: Academic Press.

PLANTINGA, H. AND DYER, C. 1990. Visibility, occlusion, and the aspect graph. *International Journal of Computer Vision 5*, 2, 137–160.

REICHERT, M. C. 1992. A two-pass radiosity method driven by lights and viewer position. Master's thesis, Program of Computer Graphics, Cornell University.

STEWART, A. J. AND GHALI, S. 1994. Fast computation of shadow boundaries using spatial coherence and backprojections. In A. GLASSNER Ed., *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series (July 1994), pp. 231–238. ACM SIGGRAPH: ACM Press. ISBN 0-89791-667-0.

STOLLNITZ, E. J., DEROSE, T. D., AND SALESIN, D. H. 1996. *Wavelets for Computer Graphics: Theory and Applications*. Morgann Kaufmann, San Francisco, CA.

STURZLINGER, W. 1994. Adaptive mesh refinement with discontinuities for the radiosity method. In *Fifth Eurographics Workshop on Rendering* (Darmstadt, Germany, June 1994), pp. 239–248.

TAMPIERI, F. 1993. *Discontinuity Meshing for Radiosity Image Synthesis*. Ph. D. thesis, Cornell University, Ithaca, NY.

TELLER, S. AND HANRAHAN, P. 1993. Global visibility algorithms for illumination computations. In *Computer Graphics Proceedings, Annual Conference Series, 1993* (1993), pp. 239–246.

TELLER, S. J. 1992. Computing the antipenumbra of an area light source. In E. E. CATMULL Ed., *Computer Graphics (SIGGRAPH '92 Proceedings)*, Volume 26 (July 1992), pp. 139–148.

TUMBLIN, J. AND RUSHMEIER, H. E. 1993. Tone reproduction for realistic images. *IEEE Computer Graphics and Applications 13*, 6 (Nov.), 42–48. also appeared as Tech. Report GIT-GVU-91-13, Graphics, Visualization & Usability Center, Coll. of Computing, Georgia Institute of Tech.

UREÑA, C. AND TORRES, J. C. 1997. Improved irradiance computation by importance sampling. In J. DORSEY AND P. SLUSALLEK Eds., *Eurographics Rendering Workshop 1997* (New York City, NY, June 1997), pp. 275–284. Eurographics: Springer Wien. ISBN 3-211-83001-4.

VEACH, E. AND GUIBAS, L. J. 1997. Metropolis light transport. In T. WHITTED Ed., *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series (Aug. 1997), pp. 65–76. ACM SIGGRAPH: Addison Wesley. ISBN 0-89791-896-7.

WALLACE, J. R., ELMQUIST, K. A., AND HAINES, E. A. 1989. A ray tracing algorithm for progressive radiosity. In J. LANE Ed., *Computer Graphics (SIGGRAPH '89 Proceedings)*, Volume 23 (July 1989), pp. 315–324.

WARD, G. 1994. A contrast-based scalefactor for luminance display. In P. HECKBERT Ed., *Graphics Gems IV*, pp. 415–421. Boston: Academic Press.

ZATZ, H. R. 1993. Galerkin radiosity: A higher order solution method for global illumination. In *Computer Graphics Proceedings, Annual Conference Series, 1993* (1993), pp. 213–220.