An exhaustive error-bounding algorithm for hierarchical radiosity

Nicolas Holzschuch, François X. Sillion

Volume 17 (*1998*), *number 4 pp. 197—218*

An exhaustive error-bounding algorithm for hierarchical radiosity

Nicolas Holzschuch[†]

François X. Sillion

*i*MAGIS[‡]

GRAVIR/IMAG - INRIA

Abstract

This paper presents a complete algorithm for the evaluation and control of error in radiosity calculations. Providing such control is both extremely important for industrial applications and one of the most challenging issues remaining in global illumination research.

In order to control the error, we need to estimate the accuracy of the calculation while computing the energy exchanged between two objects. Having this information for each radiosity interaction allows to allocate more resources to refine interactions with greater potential error, and to avoid spending more time to refine interactions already represented with sufficient accuracy.

Until now, the accuracy of the computed energy exchange could only be approximated using heuristic algorithms. This paper presents the first exhaustive algorithm to compute fully reliable upper and lower bounds on the energy being exchanged in each interaction. This is accomplished by computing first and second derivatives of the radiosity function where appropriate, and making use of two concavity conjectures. These bounds are then used in a refinement criterion for hierarchical radiosity, resulting in a global illumination algorithm with complete control of the error incurred.

Results are presented, demonstrating the possibility to create radiosity solutions with guaranteed precision. We then extend our algorithm to consider linear bounding functions instead of constant functions, thus creating simpler meshes in regions where the function is concave, without loss of precision.

Our experiments show that the computation of radiosity derivatives along with the radiosity values only requires a modest extra cost, with the advantage of a much greater precision.

1. Introduction

Global illumination algorithms now have many applications. One of the most promising fields is in urban and architectural planning, where the use of a global illumination algorithm allows to visualize a future building, and thus to check for misconceptions. For example, it becomes possible to check the ergonomy of the workplace — is there enough light, or too much? — or to ensure that the items in a museum are properly lit.

In such applications, it is vital to be able to quantify the light arriving on each point of the scene, in order to give the user a precise range in which the illumination is guaranteed to fall.

Global illumination algorithms generally have at least a parameter that the user can manipulate, choosing either fast computations or precise results. For Monte-Carlo ray tracing algorithms, this parameter can be the number of rays. For hierarchical radiosity algorithms, it can be the refinement threshold, used to decide whether or not to refine a given

† Current position: Invited Researcher, Department of Computer Science, University of Cape Town, South Africa.

‡ *i*MAGIS is a joint research project between CNRS, INRIA, INPG and Université Joseph Fourier — Grenoble I. Postal address: B.P. 53, F-38041 Grenoble Cedex 9, France. E-mail: `Nicolas.Holzschuch@imag.fr`.

interaction. Until recently, however, we had little knowledge of the total precision of the result computed, or of the relation between the parameters and this precision. Even if it was clear that spending more time on the simulation would produce more precise results, we could not quantify precisely this increase.

In 1994, Lischinski[1] proposed a refinement criterion for hierarchical radiosity such that the error on the energy at each point of the scene could be controlled by the refinement threshold. Their algorithm used upper and lower bounds on the point-to-area form factor for each interaction in order to compute upper and lower bounds for the radiosity at each point in the scene. However, they had no way to compute reliable upper and lower bounds for the point-to-area form-factor on a given interaction, and still resorted to sampling — computing a set of values for the form-factor, and taking the minimum and maximum of these values.

Although Lischinski's method is easy to implement, it is not totally reliable. In this paper, we present a method allowing to compute fully reliable upper and lower bounds for the point-to-area form-factor on any interaction. To achieve this goal, we use our knowledge of the point-to-area form-factor derivatives together with its concavity properties.

These concavity properties of the point-to-area form-factor are described in section 3. They extend the unimodality conjecture proposed by Drettakis[2, 3]. Like the unimodality conjecture, they are only conjectures, and despite their apparent simplicity, we have been unable to find a complete demonstration for them. However, we also have been unable to exhibit a counter-example.

As is explained in appendix B, we can compute exact values for the derivatives of the point-to-area form-factor; either for the first derivative, the gradient vector, or for the second derivative, the Hessian matrix. As we shall also see in appendix B, it is indeed faster to compute an exact value for the form-factor derivative than computing approximate values using several samples. Using our knowledge of the derivatives along with the concavity properties of the point-to-area form-factor, we show in section 4 how to derive bounds for the point-to-area form-factor in any unoccluded interaction. We also show an implementation of the refinement criterion using these bounds.

When dealing with partially occluded interactions we can not use the previous bounds, as the concavity conjectures do not hold in this case. But we can exhibit two emitters that are convex and bound the actual emitter, which we call the minimal and the maximal emitter. Using the previously defined algorithm, we find an upper bound for the maximal emitter, and a lower bound for the minimal emitter. The algorithm for finding these convex emitters is detailed in section 5.

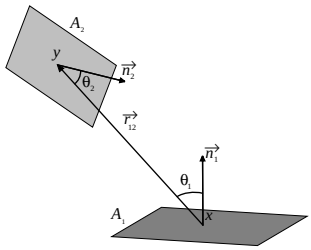


Figure 1: *Geometric notations for the radiosity equation.*

2. Background

The radiosity method was introduced in the field of light transfer in 1984 by Goral[4]. This method uses a simplification in order to solve the global illumination problem: it assumes that all the objects in the scene are ideal diffuse surfaces: their bidirectional reflectance is uniform, and thus does not depend on the outgoing direction.

In this case, the radiosity emitted at a given point x can be expressed as an integral equation:

$$B(x) = E(x) + \rho_d(x) \int_{y \in S} B(y) \frac{\cos\theta_1 \cos\theta_2}{\pi r^2} V(x,y) dy \tag{1}$$

In this equation, S is the set of all points y. r is the distance between point x and point y, θ_1 and θ_2 are the angles between the $\overrightarrow{xy}$ vector and the normals to the surfaces at point x and y respectively (refer to figure 1 for the geometric notations). $\rho_d(x)$ is the diffuse reflectance at point x, and $V(x,y)$ expresses whether point x is visible from point y or not.

In order to solve equation 1, Goral[4] suggested to discretize the scene into a set of patches $[P_i]$, over which a constant radiosity, B_i is assumed.

In this case, the radiosity at point x becomes:

$$B(x) = E(x) + \rho_d(x) \sum_i B_i \int_{y \in P_i} \frac{\cos\theta \cos\theta'}{\pi r^2} V(x,y) dy \tag{2}$$

The purely geometric quantity

$$F_i(x) = \int_{y \in P_i} \frac{\cos\theta \cos\theta'}{\pi r^2} V(x,y) dy$$

is called the point-to-area form-factor at point x from patch i. It only depends on the respective positions of point x and patch i.

Since we assume a constant radiosity value within the patch, we can compute this value as the average of all the point values. This leads to a matrix equation:

$$B_j = E_j + \rho_i \sum_i F_{ji} B_i \tag{3}$$

where the geometric quantity

$$F_{ij} = \frac{1}{A_j} \int_{x \in P_j} \int_{y \in P_i} \frac{\cos\theta \cos\theta'}{\pi r^2} V(x,y) dx dy$$

is called the form-factor. Schröder[5] showed that there is a closed form expression for the form-factor in the case of two fully visible polygonal patches. In the general case, we do not have access to the exact value of the form-factor, but only to approximate values.

Equation 3 can be solved in an iterative manner, using Jacobi or Gauss-Seidel iterative methods (see Cohen[6]). The problem is that in order to compute one full bounce of light across the surfaces in the scene, we have to compute the entire form-factor matrix, which is quadratic with respect to the number of patches.

A significant improvement over the classical radiosity method is hierarchical radiosity. In "standard" radiosity, the discretisation of one object into patches does not depend on the objects with which it interacts. In order to model the interaction between objects that are very close, and exchange lots of energy, we need to subdivide them into many patches, so as to get a precise modelling of the radiosity. On the other hand, an interaction between two objects that are far away could be modelled with fewer patches.

In hierarchical radiosity, introduced in 1990 by Hanrahan[7], each object is subdivided into a *hierarchy* of patches, with each node in the hierarchy carrying the average of the radiosity of its children. Interaction between objects far away from each other are modelled as interactions between nodes at a high level in each hierarchy. On the other hand, interactions between objects close to each other are modelled as interactions between nodes at a lower level in the hierarchy, thereby allowing more precision in the modelling of radiosity. Each interaction between two nodes is modelled by a *link*, a data structure carrying the identity of the sender and the receiver, as well as the form-factor, and possibly other informations on the respective visibility of both patches. This hierarchical radiosity algorithm has later been extended using wavelets (see Gortler[8]).

The most important step in the hierarchical radiosity method is the decision whether or not to refine a given interaction. This decision is deferred to a *refinement criterion*. Early implementations of the hierarchical radiosity method used crude approximations of the form-factor between two patches. It was known that these form-factor estimates were most imprecise when the result of the approximation was large. Hence, interactions were refined as long as the form-factor estimate was above a certain threshold (Hanrahan[7]).

This refinement criterion does not give the user a full control of the precision on the modelling of the radiosity function. In particular, it does not give any *guarantee* that it will refine all problematic interactions, and it can also refine excessively in places where the solution has already attained a correct level of precision (Holzschuch[9]).

Part of these problems can be addressed by using discontinuity meshing, where the patches are first subdivided along the discontinuity lines of the radiosity function and its derivatives (see Heckbert[10], Lischinski[11, 12] and Drettakis[13]). These discontinuity lines can be computed using geometric algorithms. However, as pointed out by Drettakis, these discontinuity lines are not of equal importance. Some of them do not have a noticeable effect on the final radiosity solution. Hence it is not necessary to compute all the discontinuity lines. Deciding which discontinuity lines are relevant is done by a refinement oracle, using heuristic methods like the one described above.

Many of the latest research results have dealt with giving the user a better control of the level of precision in the modelling of radiosity in the hierarchical radiosity method.

In the most promising paper on the subject, Lischinski[1], suggested to compute for each interaction an upper and lower bound for the point-to-patch form-factor between the points of the receiving patch and the emitting patch, namely $F_{\max}$ and $F_{\min}$, as well as an upper and lower bound for the radiosity of the emitting patch, using information already available in the hierarchy. We then know that the radiosity on the receiving patch is between $F_{\max}B_{\max}$ and $F_{\min}B_{\min}$.

Hence, the uncertainty on the radiosity on the receiving patch, due to this particular interaction is:

$$\delta B_{\text{receiver}} = F_{\max}B_{\max} - F_{\min}B_{\min}$$

The inaccuracy on the energy of the receiving patch, due to this particular interaction, is:

$$\delta E_{\text{receiver}} = A_{\text{receiver}} \left(F_{\max}B_{\max} - F_{\min}B_{\min}\right)$$

We can then decide to refine all interactions where this imprecision on the transported energy is above a given threshold. The most difficult part in this algorithm is finding reliable values for the bounds on the form-factor. Lischinski[1] suggested computing exact values for the point-to-area form factor at different sampling points on the receiver, and using the maximum and minimum value at these sampling points as the upper and lower bounds. Although this algorithm does not give totally reliable bounds, it does provide a close approximation, and is quite easy to implement on top of an existing hierarchical radiosity implementation.

In the following sections we show that it is possible to compute reliable upper and lower bounds for the point-to-area form factor. These bounds can then be used in the preceding algorithm, allowing the refinement of all interactions where the inaccuracy on the transported energy is above the threshold.

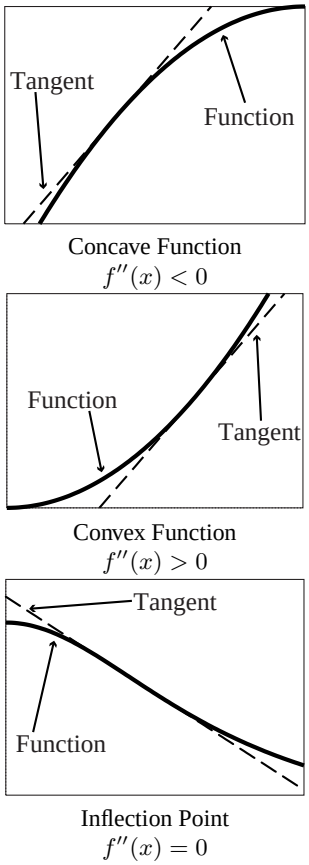


Figure 2: *Concavity for univariate functions.*

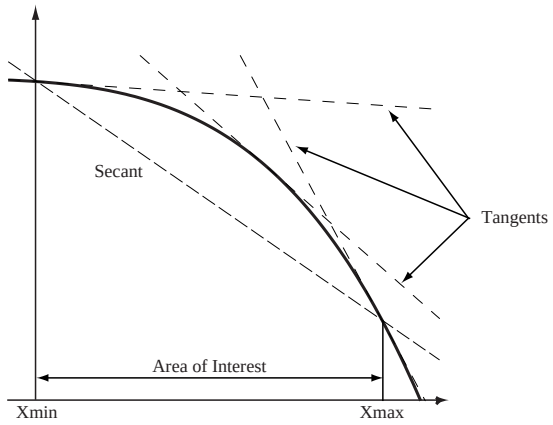


Figure 3: *A function that remains concave across an interval lies above its secant, and below all its tangents on this interval.*

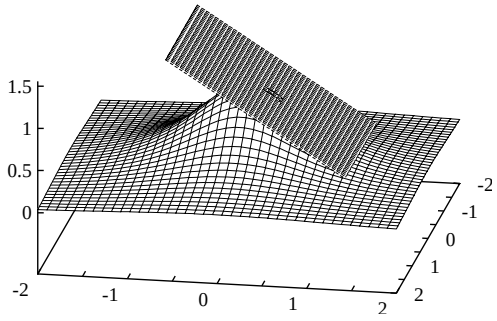


Figure 4: *A point where the function is concave: the function lies* below *the tangent plane.*

3. The Concavity Conjectures

3.1. Definition of Concavity

Univariate functions are said to be concave at a point when they lie entirely below their tangent at that point; conversely, they are said to be convex when they lie above their tangent. When the function crosses its tangent, the point is said to be an inflection point (see figure 2). Classically, the concavity of the function is linked to the sign of its second derivative: if the second derivative is positive, then the function is convex. If it is negative, then the function is concave. It is only when the second derivative changes sign that we have an inflection point.

Concavity is often used to find upper and lower bounds for functions; if a function is concave on an interval, then it is below all its tangents on this interval, and above all its secants (see figure 3). Since concavity allows bounding by affine functions (like tangents) instead of constants, it generally provides bounds that are closer to each other, and hence a "better" range.

This notion of concavity extends naturally to bivariate functions, such as radiosity defined over a surface. A bivariate function is said to be concave at a point when it lies below its tangent plane (see figure 4), convex when it lies above its tangent plane and indefinite when the function crosses the tangent plane (see figure 5). As with univariate functions, concavity can be used to find upper and lower bounds: if a function is concave over a triangular area, then on this area it lies below all its tangent planes, and above the secant plane defined by the three corners of the triangle.

A univariate function usually crosses its tangent at an isolated point, the inflection point. Contrarily, the set of points where a bivariate function crosses its tangent plane is a whole region.

The second derivative of a bivariate function is a 2×2 matrix, called the *Hessian matrix*. As with univariate functions, the concavity of the function is linked to its second derivative: if the Hessian matrix is definite positive, then the function is convex; if the Hessian matrix is definite negative, then the function is concave; if the Hessian matrix is indefinite, then the function is indefinite. The Hessian can be ex-

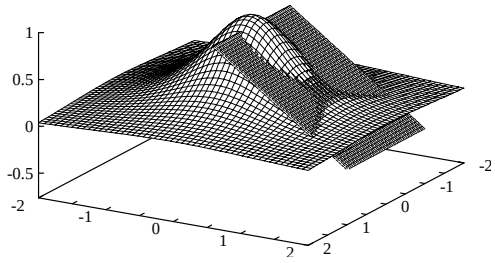


Figure 5: *A point where the concavity is indefinite: the function crosses its tangent plane.*

pressed with respect to the partial derivatives of the function:

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial u^2} & \frac{\partial^2 f}{\partial u \partial v} \\ \frac{\partial^2 f}{\partial u \partial v} & \frac{\partial^2 f}{\partial v^2} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} r & s \\ s & t \end{bmatrix} \qquad (4)$$

The Hessian is definite if $rt - s^2$ is positive. It is definite-positive if $rt - s^2$ is positive and r is positive, definite-negative if $rt - s^2$ is positive and r is negative. If $rt - s^2$ is negative or null, the Hessian is indefinite, and the function crosses its tangent plane.

It must be noted that a function is necessarily concave where it has a local maximum, and convex wherever it has a local minimum. This property is true both for uni- and bivariate functions.

3.2. Concavity of the Point-To-Area Form Factor

3.2.1. Background

Let us single out an interaction between an emitting patch and a receiving patch. We seek an upper and a lower bound for the point-to-area form-factor across the receiver. These upper and lower bounds can then be used by a refinement oracle, as introduced by Lischinski[1].

Using the algorithm described in appendix B, we have access to the form-factor and to its derivatives at any point of the receiver. However, these values are only valid at this specific point. Since we seek a result valid across the whole receiver, we must exhibit a property of the point-to-area form-factor that is valid across the receiver.

A similar approach was used by Drettakis[2, 3]. In the case of a finite convex emitter, with constant radiosity, and of an infinite receiver, Drettakis made the following two conjectures:

Conjecture U1 Radiosity on the receiver has only one maximum.

Conjecture U2 Radiosity on any line on the receiver has only one maximum.

These two conjectures are referred to below as the *unimodality conjectures*.

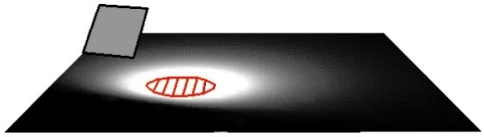


Figure 6: *The C1 conjecture: the radiosity function has indefinite concavity everywhere, except over a convex area (hatched), where the radiosity function is concave.*

3.2.2. Concavity Conjectures

Like Drettakis, we consider a finite convex emitter, with constant radiosity, and we assume the receiver is an infinite plane. We state the following two conjectures on the concavity of the radiosity on the receiver:

Conjecture C1 The Hessian matrix of the radiosity function is indefinite everywhere, except over a bounded area. On this area, the radiosity function is concave. Furthermore, the area is convex.

Conjecture C2 On any line drawn on the receiver, radiosity is concave over a bounded interval, and convex everywhere else.

Figure 6 illustrates the C1 conjecture: the radiosity function is indefinite everywhere — and crosses its tangent plane — except over a convex region (hatched).

Figure 7 illustrates the C2 conjecture: the radiosity function defined over a given line is convex across $[-\infty, a]$ and across $[b, +\infty]$, and concave across $[a, b]$.

Despite their apparent simplicity, these conjectures have yet escaped demonstration. It is obvious that they are true in the simplest case of a point light source sending light in all directions. However, even for the case of a differential emitter area instead of a point light source, it has not been possible so far to prove the concavity conjectures. Appendix A is a detailed study of the differential emitter area.

3.2.3. Relationship between the conjectures

Our concavity conjectures are actually an extension of the unimodality conjectures: that is, C1 implies U1, and C2 implies U2. Note that we also know that U2 implies U1:

$$\begin{cases} \text{U2} \implies \text{U1} \\ \text{C2} \implies \text{U2} \\ \text{C1} \implies \text{U1} \end{cases}$$

U2 $\implies$ U1:

Proof Assume U1 is false. Then there exists at least two maxima for the radiosity function, M1 and M2. On the line joining M1 and M2 there are two maxima, which is in contradiction with U2. □

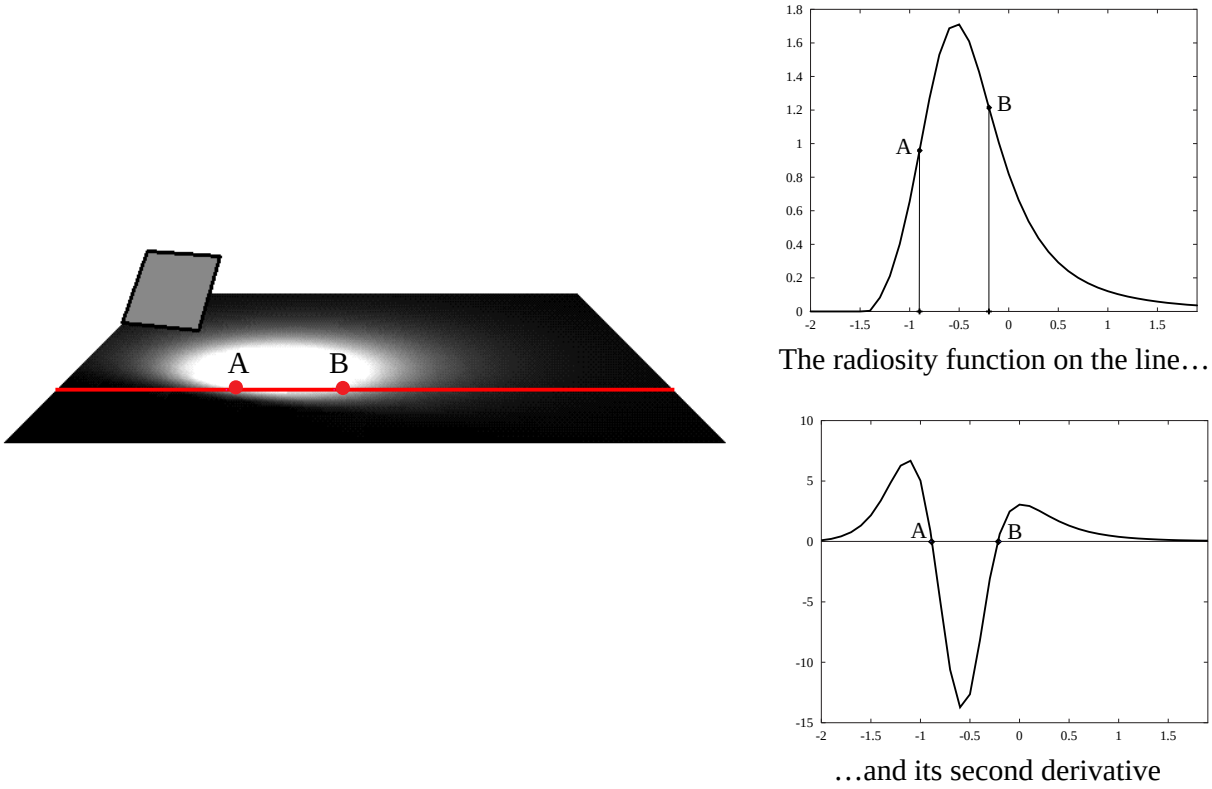


Figure 7: *The C2 conjecture: the radiosity function on a line is concave only over a finite interval, [AB].*

C2 $\Longrightarrow$ U2:

Proof The function is concave on the neighbourhood of each local maximum. If there are two local maxima on a line, there must be a local minimum between them. In the neighbourhood of this local minimum, the function would have to be convex, which is impossible because of C2. □

C1 $\Longrightarrow$ U1:

Proof Assume U1 is false. Then there exists at least two local maxima for the radiosity function. On the neighbourhood of each maximum, the radiosity function is concave. But between the two maxima, there must be a pass-like point, where the concavity is indefinite. This is in contradiction with C1. □

No relationship between C1 and C2: An important point is the *independence* of our two concavity conjectures. C1 does not imply C2, and C2 does not imply C1.

4. Error Control for Unoccluded Interactions

In this section, we describe our algorithm for finding upper and lower bounds for the point-to-area form-factor across the receiver. These values are then used by a refinement oracle like the oracle introduced by Lischinski[1].

4.1. Computing Radiosity Derivatives

Let us call A_2 the emitting patch, A_1 the receiver and x a point on the receiver (see figure 1). In this case, there is an exact formula for the point-to-area form factor (Siegel and Howell[14]):

$$F(x) = -\frac{1}{2\pi}\vec{n}_1 \cdot \oint_{\partial A_2} \frac{\vec{r}_{12}}{\|\vec{r}_{12}\|^2} \times d\vec{\ell}_2 \qquad (5)$$

where the integral is on ∂A_2, the contour of A_2, and $d\vec{\ell}_2$ is the differential element of this contour.

Using this expression of the point-to-area form-factor, it is possible to compute exact formulae for both its first and second derivatives. These formulae for the derivatives are easily implemented, giving access to exact values for the function and its derivatives (see appendix B, and also Arvo[15] or Holzschuch[16, 17]).

If we compute simultaneously the point-to-area form-factor and its derivatives, we can save computation time by reusing some geometric quantities that appear in several formulae. In this case, the overall cost of computing the derivatives is reasonable: there is an increase of 40% for computing the gradient along with the form-factor, and an increase of 100% for computing both the gradient and the Hes-

sian matrix (see appendix B and Holzschuch[16, 17]). This cost must be balanced against what it would require to compute approximate values for the derivatives using several form-factor computations: in this case, the cost increase for the gradient would be of 100%, and that of the Hessian 600%.

In our refinement phase, we compute the values of the point-to-area form-factor and its derivatives at the vertices of the receiving patch. These values can be reused in the radiosity propagation phase to obtain the radiosity values at the vertices.

4.2. Computing Bounds for the Point-to-Area Form-Factor

We show here how our knowledge of the point-to-area form-factor and its derivatives at the vertices of the receiving patch, used jointly with our conjectures, gives us access:

- first, to the *location* of the maximum and the minimum of the point-to-area form-factor,
- second, to an exact value for the minimum,
- third, to an upper bound for the maximum.

4.2.1. The Minimum is at one of the Vertices

An immediate consequence of the unimodality conjectures (U1 and U2) is that the minimum for the point-to-area form-factor is necessarily at one of the vertices of the receiver:

- If the minimum was inside the receiving patch, A1, then there would exist several local maxima for the point-to-area form-factor on the plane supporting A1 — this is in contradiction with U1. Hence, the minimum across A1 must be on the contour of A1.
- The contour of A1 is made of polygonal edges. If on one of these edges the minimum is inside the edge then on the line supporting the edge the form-factor must have two maxima — this is in contradiction with U2.
- Hence, the minimum can only be at one of the vertices of A1.

4.2.2. An exact value for the minimum

Since we chose to compute the point-to-area form-factor at the vertices of the receiving patch, A1, we do have access to the exact value of the minimum across A1: it is the minimum of our computed values for the point-to-area form-factor at the vertices of A1.

4.2.3. Finding the Position of the Maximum

A consequence of U2 is that given a point x, given the point-to-area form-factor $F(x)$ and its gradient at point x, $\nabla F(x)$, for all points p such that $\overrightarrow{xp} \cdot \nabla F(x) < 0$, we have $F(p) < F(x)$.

Otherwise, there would be one local minimum between p and x on the line passing through p and x, and hence two local maxima, which is in contradiction with U2.

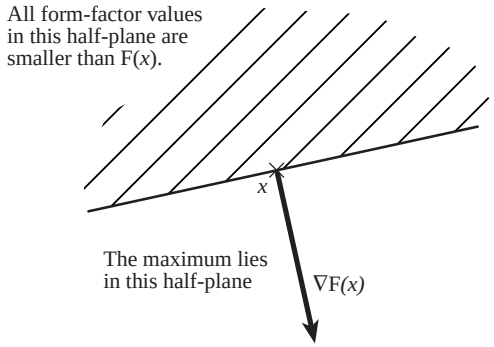


Figure 8: *Knowledge of the gradient helps find the position of the maximum.*

Hence the maximum of the point-to-area form-factor can only be in the half-plane defined by: $\overrightarrow{xp} \cdot \nabla F(x) \geq 0$ (see figure 8.)

This property gives us an algorithm to determine whether the maximum for the point-to-area form-factor across the receiving patch A1 can lie *inside* the patch, or if it must be at one of the vertices (see figure 9):

- For each vertex, there is a half-plane (defined by the form-factor gradient at this vertex) where the form-factor value can be greater than the value at the vertex.
- The intersection of these half-planes is an area where the point-to-area form-factor value can be greater than the value at all the vertices. The intersection of this area with the receiving patch is either empty or not empty.
- If this intersection with the patch is not empty, then there exists an area inside the patch where the maximum can be.
- If this intersection is empty, then the maximum for the form-factor across the patch must be at one of the vertices.

4.2.4. If the Maximum is at one of the Vertices

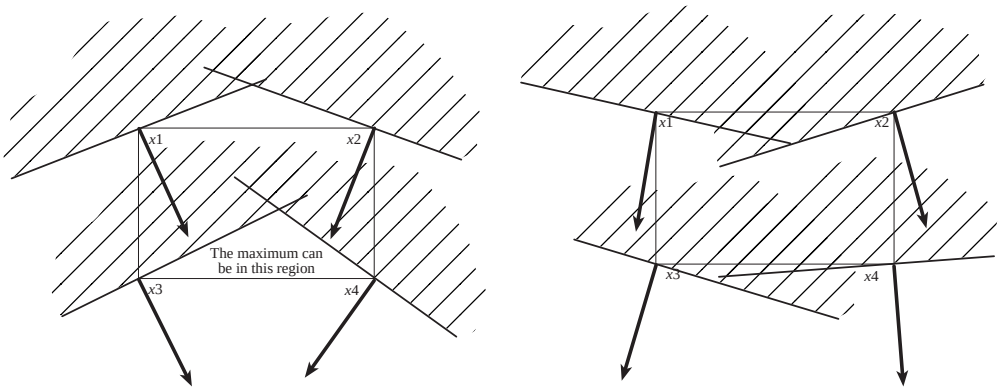
If the above algorithm tells us that the maximum can only be at one vertex of the receiving patch, then we know the exact value of the maximum: it is the value of the point-to-area form-factor at that vertex.

4.2.5. If the Maximum is Inside the Receiving Patch

If the above algorithm tells us there exists an area inside the receiving patch A1 where the maximum can be, then we do not have access to the exact value of the maximum of the point-to-area form-factor across A1.

The only thing we know at this stage is that the value of the maximum must be greater than the values computed at the vertices of A1.

There are three kind of algorithms for finding an upper bound for the point-to-area form-factor across A1:



The maximum can be inside the polygon. The maximum is at one of the vertices.

Figure 9: *Using the gradient to locate the maximum inside or outside the receiving patch.*

Heuristic Algorithms: Compute another sample value for the point-to-area form-factor inside patch A1. The position of the sampling point can be arbitrary or can make use of the information given by the form-factor gradient.

Concavity Algorithms: If the point-to-area form-factor function on the receiving patch is concave, we use the tangent planes to find an upper-bound.

Geometric Algorithms: Using geometric tools, build an emitter that encloses the actual emitter for all the points of the receiving patch, and for which we can find the value of the maximum. This value is an upper-bound.

Heuristic algorithms include gradient descent algorithms, as described by Arvo[15] and Drettakis[2, 3]. Gradient descent algorithms make use of the information provided by the gradient to subdivide the receiving patch until convergence. The gradient can either be approximated (Drettakis[2, 3]) or an exact value (Arvo[15]).

In our implementation, we use concavity algorithms wherever possible, and resort to geometric algorithms if the point-to-area form-factor function is not concave.

4.2.5.1. Concavity Algorithms According to C1, the zone where the point-to-area form-factor function is concave is a convex one. As a consequence, if the form-factor Hessian is definite negative at the vertices of the receiving patch, then it stays definite negative across the receiving patch.

In this case, the form-factor function lies below all its tangent planes at the vertices across the receiving patch. We know these tangent planes since we know the form-factor gradient at the vertices. Finding an upper bound for the point-to-area form-factor is then equivalent to computing the intersection of the tangent planes.

This is mainly a linear programming problem (see, for example, Preparata[18]); the computational complexity of the problem depends on the dimension of the problem — which here is always two since we are dealing with bivariate functions — and on the number of vertices in the receiving patch. Usually, in hierarchical radiosity algorithms, we are restricting ourselves to triangular or quadrangular patches. If this is the case, we can assume the complexity of computing the intersection of the tangent planes is constant.

4.2.5.2. Geometric Algorithms If the form-factor Hessian is not definite negative at all the vertices of the receiving patch, then the point-to-area form-factor function is not concave across the entire receiving patch. It is therefore not possible to use concavity algorithms. In this case, we resort to geometric algorithms: in a plane parallel to the plane of the receiver, we construct an emitter with the following two properties:

- From all the points of the receiver, it is seen as including the original emitter.
- It has two axes of symmetry, so that we can find the maximum form-factor due to the emitter.

The reason for the second item lies in the symmetry principle: if the emitter and the receiver are left unchanged by a planar symmetry, then so is the point-to-area form-factor function on the receiver; thus its maximum can only lie on the intersection of the plane of the symmetry and of the plane of the receiver. If there are two planes that leave the emitter and the receiver un-changed, then the maximum can only be at their intersection (see figure 24, in the color section).

To build this emitter:

- select a plane P parallel to the plane of the receiving patch;
- for each vertex V_i of the receiving patch, build the projection p_i of the original emitter according to this vertex on P (see figure 10);
- this projection is totally equivalent to the original emitter for this particular vertex;

- any convex region enclosing all the p_i projections is seen from all the points of the receiver as enclosing the original emitter; as a consequence, the point-to-area form-factor due to this convex region is greater than the point-to-area form-factor due to the actual emitter;
- building a convex region enclosing the p_i is a standard geometry problem (see, for example, Foley *et al.*[19], Kay[20] or Toth[21]). Constraining this convex region to have two axes of symmetry can either be a consequence of the bounding object used, like ellipses and rectangles, or be a property we add afterward. Since our problem is a two dimensional geometry problem — although we have a set of three dimensional data points — we start by projecting our p_i onto one of the coordinates planes (x, y), (z, x) or (y, z). Once we have built the result in this coordinate plane, we will project it back onto the emitter plane.
- Several algorithms can be used, either giving a faster result, but a greater enclosing emitter, and hence a greater upper-bound, or requiring more time, but giving an enclosing emitter that is closer to the p_i, and hence a smaller upper-bound:
  - Build the convex hull of the p_i, then build a region with two axes of symmetry enclosing the convex hull. This gives the smaller enclosing emitter, but requires more computation time
  - Build a bounding rectangle enclosing the p_i inside the emitter plane, as in Toth[21]. This is one of the fastest possible algorithm. Furthermore, it naturally gives an enclosing emitter with two axes of symmetry, so there is no construction time involved for building the symmetries.
  - Build a bounding ellipse enclosing the p_i inside the emitter plane. This algorithm is slower, but it also gives an enclosing emitter with two axes of symmetry, so there is no construction time involved for building the symmetries.
  - A bounding rectangle using the (x, y, z) axes can give an enclosing emitter much bigger than the p_i, thus inducing a greater upper bound. A simple improvement is to use *slabs*, as suggested in Kay[20]. In this case, in order to build an object with two axes of symmetry, we have to restrict ourselves to two sets of orthogonal slabs. This algorithm requires more computational time than the previous algorithm, but can give a significantly smaller enclosing emitter.
  - If n_e is the number of vertices of the emitter, and n_r the number of vertices of the receiver, the total number of vertices for all the p_i is $n_e n_r$. In this case, the complexity of the convex hull algorithm is $O(n_e n_r \log n_e n_r)$, and the complexity of the other three algorithms is $O(n_e n_r)$.

Figure 11 gives an example of the construction of an enclosing emitter.

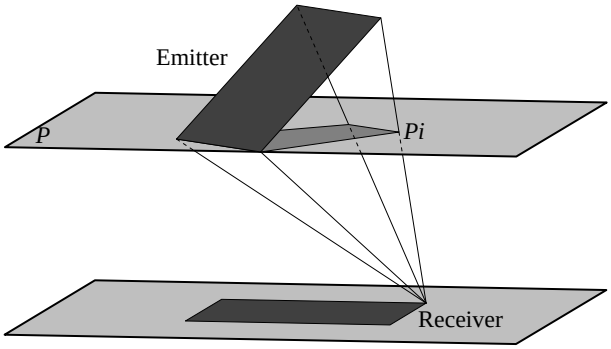


Figure 10: *The projection of the emitter on the plane P from a given vertex.*

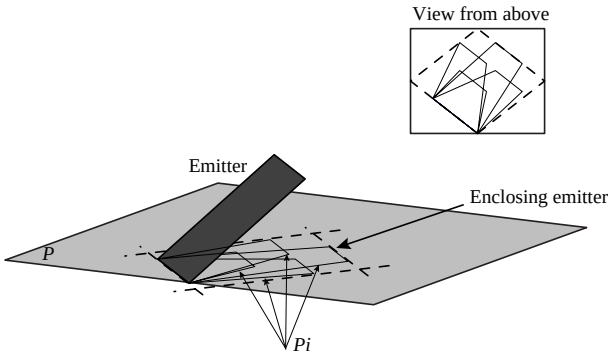


Figure 11: *Building an enclosing emitter in order to find an upper bound.*

4.3. Implementation and Testing

4.3.1. Refinement Criterion

Once we have access, for each iteration, to the minimum and maximum form-factor, it is possible to implement a refinement criterion based on their difference. Following the algorithm suggested by Lischinski[1], we refine every interaction such that:

$$A_{\text{receiver}} \left(B_{\max} F_{\max} - B_{\min} F_{\min}\right) > \varepsilon$$

This means that we refine an interaction whenever the uncertainty on the incoming energy of the receiving patch is above the threshold ε.

4.3.2. Resulting Mesh Simplification

In regions where the Hessian matrix of the form-factor is definite-negative, we know that the form-factor can be bounded between the tangent planes and the secant planes. We can use these bounding planes to find tighter upper and lower bounds for the form-factor.

The form-factor for all the points on the receiving patch

lies below all the tangent planes for the point-to-area form factor, and above the secant plane. Therefore, we can say that our uncertainty on the point-to-area form-factor on the receiver is equal to the maximum of the distance between the secant plane and these tangent planes.

Computing this distance is again a linear programming problem (see, for example, Preparata[18]). The complexity depends on the number of vertices of the receiver, n_r, which is usually three or four. Let us denote by E_{FF} this uncertainty on the form-factor. E_{FF} can be used in our expressions as a replacement for $F_{\max} - F_{\min}$. Using the fact that:

$$(B_{\max}F_{\max} - B_{\min}F_{\min}) = \\ B_{\max}(F_{\max} - F_{\min}) + F_{\min}(B_{\max} - B_{\min})$$

we decide to refine a given interaction if

$$A_{\text{receiver}}\,(B_{\max}E_{FF} + F_{\min}(B_{\max} - B_{\min})) > \varepsilon$$

It must be noted that this new bounding of the form-factor does not introduce any uncertainty. We are still bounding the form-factor by fully reliable functions. However, since these functions are affine instead of constants, they provide much tighter bounds, and we can expect a simpler mesh in the areas where the point-to-area form factor is concave.

Figure 25 (in the color section) shows the result of our refinement criterion on a simple box, with only direct illumination. Notice that the mesh produced is coarser in some areas with respect to the immediately neighbouring areas (the disc-shaped area on the floor, and the drop-shaped areas on the walls). These are the places where the Hessian is definite-negative.

This refinement criterion extends, in some ways, the mesh simplification found in previous work (Holzschuch[9]). The shape of the mesh produced is quite similar between our new algorithm and the algorithm in Holzschuch[9]. However, our new refinement criterion, while keeping low memory costs, also gives fully reliable upper and lower bounds on the radiosity of each patch.

4.3.3. Dealing with Singularities

4.3.4. Relative Complexity of the Algorithm

Our algorithm requires the computation of the first two derivatives of the point-to-area form-factor at the vertices of the receiver. This implies a 100 % increase on the computation time for each vertex (see appendix B). That is to say, computing the point-to-area form-factor and its derivatives costs twice what it would cost to compute the point-to-area form-factor alone.

Since vertices are shared by several patches, this overhead cost is shared by several interactions. On the average, we are only computing one point-to-area form-factor and its derivatives for each patch. Thus, the cost of our algorithm is approximately the cost of computing two point-to-area form-factors for each patch, plus the time needed for the exploitation of the derivatives for computing upper and lower bounds.

Existing heuristic refinement algorithms (see Lischinski[1]) compute one form-factor sample for each of the receiver vertices, plus one sample at the center of the receiving patch. If we assume that the form-factor values at the vertices are shared with the neighbouring patches, we are computing an average of two point-to-area form-factors for each receiver.

Thus, the cost of the heuristic algorithm and the cost of our algorithm are roughly similar. The main overhead of our algorithm when compared with the heuristic algorithm is the time needed for the actual computations for finding the position of the maximum and for finding an upper bound for the maximum, when necessary.

Hence, the relative costs of our refinement criterion are in fact quite small and can be generally regarded as acceptable, especially with respect to the complete control it gives on the error carried by each interaction.

Also, our algorithm allows for a significant mesh simplification (see figure 25, in the color section) which may, depending on the scene considered, induce a smaller computation time for the exhaustive refinement criterion when compared to a heuristic refinement criterion.

5. Error Control for Partially Occluded Interactions

The above algorithm for finding upper and lower bounds only works in the case of unoccluded interactions, and with a convex emitter. This algorithm relies on the concavity and unimodality conjectures, which do not hold if there are occluders between the emitter and the receiver.

However, it is possible to construct, using geometrical tools, a minimal and a maximal emitter that have the following qualities:

- both are convex;
- any point of the minimal emitter is fully visible from the receiver;
- the maximal emitter contains all the points of the emitter that are visible from at least one point of the receiver;

Then at any given point on the receiver,

- the form-factor due to the minimal emitter is lesser or equal to the actual form factor,
- and the form-factor due to the maximal emitter is greater or equal to the actual form-factor.

We apply our previous algorithm to these emitters, and find a lower bound using the minimal emitter, and an upper bound using the maximal emitter.

Figure 26 (in the color section) shows an example of minimal and maximal emitters for a simple configuration with only one occluder: the small red square on the ground is the

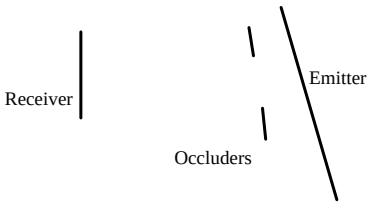


Figure 12: *A single interaction with occluders.*

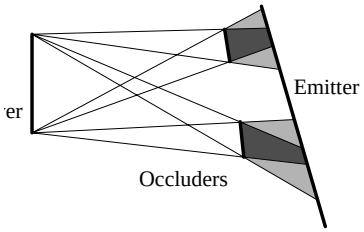


Figure 13: *Computing the "umbra" and "penumbra" volumes using the receiver as a light source.*

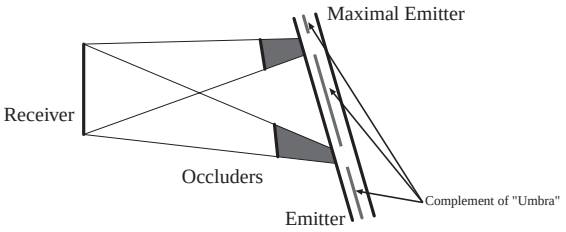


Figure 14: *The maximal emitter can be any convex including the complement of the "umbra" region.*

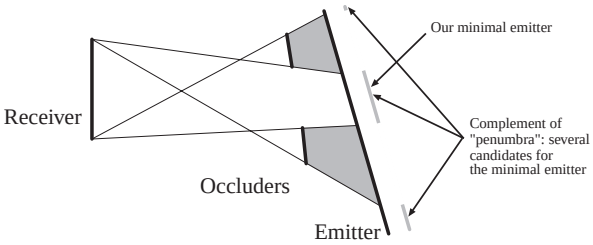


Figure 15: *Several possible candidates for the minimal emitter.*

receiver; the black square with a white border is the occluder, and the bright red area is the minimal emitter — the part of the emitter that is visible from all the points of the receiver. The dark red area is the maximal emitter. The blue line is the contour of the emitter as it is seen from one of the points of the receiver.

5.1. Computing the minimal and maximal emitter

Our definition of minimal and maximal emitter bears a strong resemblance with the definition of umbra and penumbra, except that the roles of the emitter and the receiver are *reversed*.

A similar algorithm has been used by Teller to computer the antipenumbra of an area light source[22], and to solve the visibility problem in a hierarchical radiosity algorithm[23], and by Drettakis[13]. Drettakis[13] used a specific data structure, the *backprojection*, which gives to the program the structure of the projection of the occluders on the emitter plane, from any point on the receiver.

Algorithms used for computing umbra and penumbra can be quickly adapted in order to compute the minimal and maximal emitter for each receiver. Let us consider a single interaction, with one emitter, one receiver, and occluders (see figure 12). We compute the umbra and the penumbra volume using the receiver as a light source (see figure 13). The intersection of these volumes with the emitter plane is a close indication of where the minimal and maximal emitter are.

5.1.1. Computing the maximal emitter using the "umbra" volume

The intersection of the emitter with the "umbra" volume is the set of points on the emitter that are totally invisible from the receiver.

The complement of this intersection is the set of points on the emitter that are visible from at least one point on the receiver.

Since our criterion only works for convex emitters, we have to build a convex emitter that includes this complement. Our basic rule is that we must not under-estimate the point-to-area form-factor, only possibly over-estimate it. Hence, the maximal emitter must be any convex region including the previously computed complement — for example the convex hull of the complement, or the bounding-box of the complement (see figure 14).

5.1.2. Computing the minimal emitter using the "penumbra" volume

Similarly, the intersection of the emitter with the "penumbra" volume is the set of points on the emitter that are at least partially hidden from the receiver.

The complement of this intersection is the set of points on the emitter that are visible all the points of the receiver.

Any convex region that is included in this complement is a suitable candidate for the minimal emitter (see figure 15).

Depending on the position of the occluders, it is possible

to have several candidates for the minimal emitter. Ideally, we would like to pick the candidate that gives the largest estimate for the minimum, since this would give tighter bounds, and hence reduce the number of un-necessary refinements. However, it is impossible to find this without computing the point-to-area form-factor for all the candidates, which would prove very time-consuming. In our implementation, we choose the candidate with the largest area, since it is likely to induce a larger form-factor.

5.2. Implementation and testing

We have implemented our algorithm for finding upper and lower bounds for the point-to-area form-factor using the maximal and minimal emitter.

Figure 27 (in the color section) shows the result of our refinement criterion on a simple scene, with a single occluder. Notice that the algorithm detects the shadow boundaries and refines properly in order to model them. Outside of the shadow, the mesh produced is identical to the mesh produced without occluders.

5.3. Complexity of the Algorithm and Possible Improvements

Our algorithm relies on computation of the umbra and penumbra volumes for all the interactions. This computation can be quite costly, if it is implemented in a naive way.

Previous work by Chin[24] has shown that the use of a BSP-tree can greatly improve the computation of umbra and penumbra volumes. Teller[22] showed that by extending the data structure used to store the interaction between patches to also store the possible occluders for this interaction, the complexity of visibility computations could be greatly reduced. Both these improvements work with our algorithm.

Our algorithm can also be used in a combination with standard discontinuity meshing, as described in Lischinski[11]. A preliminary light-source discontinuity meshing will reduce the complexity of the minimal and maximal emitter computations by providing occlusion information and reducing the number of patches where we have to compute these emitters.

The backprojection algorithm described by Drettakis[13, 3] gives for each patch created during the discontinuity meshing step the geometric structure of the emitter as seen from this patch. Implementing our algorithm on top of a backprojection algorithm should be a straightforward post-processing step.

It has been shown (Lischinski[11] and Drettakis[13, 3]) that the boundary of the umbra volume can include a quadric surface, and hence can be quite complex to model. However, our algorithm does not require a complete computation of the umbra and penumbra volumes for each interaction, but only the computation of a surface included in the umbra volume, and of a surface enclosing the penumbra volume. Two such surfaces can be computed in a straightforward way:

- For each occluder:
  - For each receiver vertex, compute the projection of the occluder onto the emitter supporting plane;
  - The intersection of these projections is the umbra volume for this particular receiver;
  - The convex hull of these projections is the penumbra volume for this receiver.
- The union of the penumbra volumes for all occluders is the penumbra volume for the entire interaction.
- The union of the umbra volumes for all occluders is not equal to the umbra volume for the entire interaction. However, it is included into the actual umbra volume (see Lischinski[11]). Hence, we can use it for building the maximal emitter.

The computation of the projection of the occluders onto the emitter supporting plane, and the computation of the union of these projections can be reused for computing the exact value of the point-to-area form-factor in the radiosity propagation phase.

The only extra cost of our refinement criterion is then the computation of the minimal and maximal emitter knowing the projection of all the occluders on the emitter plane. This is a two-dimensional problem, computing a convex region that contains the complement of the umbra volume, and another convex region that is included into the complement of the penumbra volume. Note that we do not have to explicitely construct the umbra and the penumbra volume, only the two convex regions. We can use several methods for computing these convex regions, as described in section 4.2.5.2. The cost of our algorithm is the cost of finding two convex regions enclosing $n_r n$ polygons, where n is the number of occluders, and n_r is the number of vertices of the receiver.

The heuristic algorithm described by Lischinski[1] uses the same computation of the exact values of the point-to-area form-factor at the vertices of the receiver, which will be reused in the radiosity propagation phase, plus the computation of the point-to-area form-factor at the center of the receiving patch, which implies the projection of the occluders on the emitter supporting plane and the computation of the union of these projections. Hence, the cost of the heuristic algorithm is n projections and the union of n two-dimensional polygons.

6. Conclusions and Future Directions

We have introduced a new and reliable way of computing the maximum and the minimum of the point form-factor on any interaction. These bounds on the form-factor allow a control of the precision of the hierarchical radiosity algorithm,

precision that can be required for certain applications of the algorithm, such as architectural planning.

These bounds have been integrated in a new refinement criterion for hierarchical radiosity. We have also presented another refinement criterion that, while maintaining control on the upper and lower bounds of the energy transported, allows a coarser mesh to be constructed in some places, thus reducing memory and computation costs.

This algorithm is a significant step in error-control for global illumination methods. Although it has been devised and implemented in a hierarchical radiosity framework, nothing in the algorithm prevents the refinement criterion to be implemented with progressive refinement radiosity, as described by Cohen[25].

Knowledge of the error produced in all the parts of the algorithm allows global illumination programs to concentrate their work on parts of the scene where the error is still large, and to skip parts where it can be neglected. Thus, our algorithm can be hoped to accelerate global illumination computations by reducing the amount of unnecessary refinement.

Our algorithm relies on several conjectures: the unimodality conjectures (U1 and U2) and the concavity conjectures (C1), as well as on a knowledge of the radiosity derivatives. Table 1 recalls, for each part of the algorithm, which conjecture and which derivatives are being used.

The concavity and unimodality conjectures assume that radiosity on the emitter is constant, that the receiver is diffuse and that there is full visibility. An extension of our error-control algorithm to cases where radiosity on the emitter is not constant, or to reflectance functions that are not constant would first require a careful study of to what extent do our concavity or unimodality conjectures still hold. For example, it is clear that they cannot hold for whatever distribution of radiosity on the emitter, but only for specific cases. These specific cases, once identified, can be used as a functional basis for radiosity.

We have dealt with the partial visibility problem by computing maximal and minimal emitter, thereby reducing the problem to two full visibility problems. However, it is known that it is possible to compute the radiosity gradient in presence of occluders (see Arvo[15]), and it seems possible to compute the radiosity Hessian in presence of occluders as well (see Holzschuch[17]). In this case, it would be possible to extend our refinement criterion to some partially visible interactions without having to compute the maximum and minimum emitter. Once again, this can be done only in specific configurations where the concavity or unimodality conjectures still hold. This is not the case for generic occluders (see figure 28, in the color section), but only for certain specific, simple occluders (see figure 29 in the color section).

Although the algorithm described in this paper makes use of the U1, U2 and C1 conjectures, and of the form-factor gradient and Hessian, table 1 shows that it is possible to build a simpler algorithm to find upper and lower bounds by using only U1, U2 and the form-factor gradient.

This algorithm would be very similar to the gradient-descent algorithms described by Arvo[15] and Drettakis[2, 13]. The main difference would be the use of geometric tools, as described in section 4.2.5.2 to find an upper bound. These geometric tools will provide a fully reliable upper bound on the receiving patch.

This simpler algorithm would not allow mesh simplification as described in section 4.3.2; also, since this simpler algorithm would only use geometric methods to find upper bounds it can be expected that it will give greater upper bounds, and hence induce more refinement than our current algorithm. On the other hand, this algorithm would not require the computation of the form-factor Hessian, thus saving computation time, and would probably be easier to extend to partial visibility cases, where C1 may not hold.

Future work will include an implementation of this simpler algorithm, and timing and memory costs comparisons between our full algorithm, the simpler algorithm and the heuristic algorithm, as well as error measurements.

7. Acknowledgements

The first author has been funded by an AMN grant from Université Joseph Fourier from 1994 to 1996.

References

1. D. Lischinski, B. Smits, and D. P. Greenberg, “Bounds and Error Estimates for Radiosity”, in *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH ’94 Proceedings)*, pp. 67–74, (1994).

2. G. Drettakis and E. Fiume, “Accurate and Consistent Reconstruction of Illumination Functions Using Structured Sampling”, in *Computer Graphics Forum (Eurographics ’93)*, vol. 12, (Barcelona, Spain), pp. C273–C284, (September 1993).

3. G. Drettakis, “Structured Sampling and Reconstruction of Illumination for Image Synthesis”, CSRI Technical Report 293, Department of Computer Science, University of Toronto, Toronto, Ontario, (January 1994).

4. C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, “Modelling the Interaction of Light Between Diffuse Surfaces”, in *Computer Graphics (ACM SIGGRAPH ’84 Proceedings)*, vol. 18, pp. 212–222, (July 1984).

5. P. Schröder and P. Hanrahan, “On the Form Factor Between Two Polygons”, in *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH ’93 Proceedings)*, pp. 163–164, (1993).

Parts of the algorithm	Conjectures required	Derivatives required
Position and value of the minimum	U1 and U2	None
Position of the maximum	U2	Gradient
Using tangents to find an upper bound	C1	Hessian
Using geometric algorithms to find an upper bound	None	None
Simplification of the mesh	C1	Hessian

Table 1: *Dependencies for the different parts of the algorithm*

6. M. Cohen and D. P. Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environments", in *Computer Graphics (ACM SIGGRAPH '85 Proceedings)*, vol. 19, pp. 31–40, (August 1985).

7. P. Hanrahan, D. Salzman, and L. Aupperle, "A Rapid Hierarchical Radiosity Algorithm", in *Computer Graphics (ACM SIGGRAPH '91 Proceedings)*, vol. 25, pp. 197–206, (July 1991).

8. S. J. Gortler, P. Schröder, M. F. Cohen, and P. Hanrahan, "Wavelet Radiosity", in *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pp. 221–230, (1993).

9. N. Holzschuch, F. Sillion, and G. Drettakis, "An Efficient Progressive Refinement Strategy for Hierarchical Radiosity", in *Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 343–357, (June 1994).

10. P. Heckbert, "Discontinuity Meshing for Radiosity", in *Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 203–226, (May 1992).

11. D. Lischinski, F. Tampieri, and D. P. Greenberg, "Discontinuity Meshing for Accurate Radiosity", *IEEE Computer Graphics and Applications*, **12**(6), pp. 25–39 (1992).

12. D. Lischinski, F. Tampieri, and D. P. Greenberg, "Combining Hierarchical Radiosity and Discontinuity Meshing", in *Computer Graphics Proceedings, Annual Conference Series, 1993 (ACM SIGGRAPH '93 Proceedings)*, pp. 199–208, (1993).

13. G. Drettakis and E. Fiume, "A Fast Shadow Algorithm for Area Light Sources Using Backprojection", in *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pp. 223–230, (1994).

14. R. Siegel and J. R. Howell, *Thermal Radiation Heat Transfer, 3rd Edition*. New York, NY: Hemisphere Publishing Corporation, (1992).

15. J. Arvo, "The Irradiance Jacobian for Partially Occluded Polyhedral Sources", in *Computer Graphics Proceedings, Annual Conference Series, 1994 (ACM SIGGRAPH '94 Proceedings)*, pp. 343–350, (1994).

16. N. Holzschuch and F. Sillion, "Accurate Computation of the Radiosity Gradient for Constant and Linear Emitters", in *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)* (P. M. Hanrahan and W. Purgathofer, eds.), (New York, NY), pp. 186–195, Springer-Verlag, (1995).

17. N. Holzschuch, *Le Contrôle de l'Erreur dans la Méthode de Radiosité Hierarchique (Error Control in Hierarchical Radiosity)*. Ph.D. thesis, Équipe iMAGIS/IMAG, Université Joseph Fourier, Grenoble, France, (March 5th, 1996).

18. F. P. Preparata and M. I. Shamos, *Computational Geometry – An Introduction*. New York: Springer Verlag, (1985).

19. J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics, Principles and Practice, Second Edition*. Reading, Massachusetts: Addison-Wesley, (1990).

20. T. L. Kay and J. T. Kajiya, "Ray tracing complex scenes", *Computer Graphics*, **20**(4), pp. 269–276 (1986). Proceedings of SIGGRAPH '86 in Dallas (USA).

21. D. L. Toth, "On ray-tracing parametric surfaces", *Computer Graphics*, **19**(3), pp. 171–179 (1985). Proceedings SIGGRAPH '85 in San Francisco (USA).

22. S. J. Teller, "Computing the antipenumbra of an area light source", in *Computer Graphics (ACM SIGGRAPH '92 Proceedings)*, vol. 26, pp. 139–148, (July 1992).

23. S. Teller and P. Hanrahan, "Global Visibility Algorithms for Illumination Computations", in *Computer Graphics Proceedings, Annual Conference Series,*

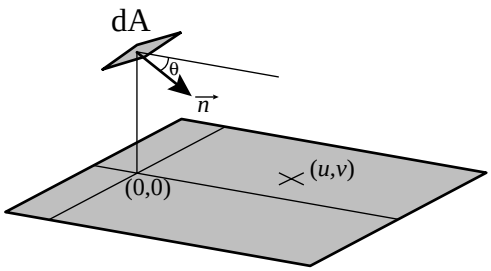


Figure 16: *A differential area emitter and an infinite receiving plane.*

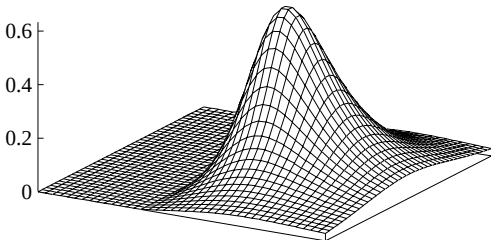


Figure 17: *An example of the point-to-area form-factor function ($\theta = \frac{\pi}{6}$).*

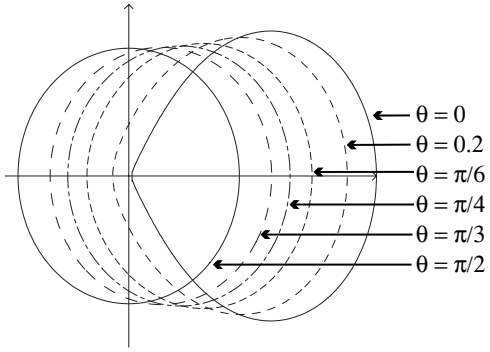


Figure 18: *The areas where the point-to-area form-factor function is concave for different values of θ.*

1993 (ACM SIGGRAPH '93 Proceedings), pp. 239–246, (1993).

24. N. Chin and S. Feiner, "Fast object-precision shadow generation for areal light sources using BSP trees", *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, **25**(2), pp. 21–30 (1992).

25. M. Cohen, S. E. Chen, J. R. Wallace, and D. P. Greenberg, "A Progressive Refinement Approach to Fast Radiosity Image Generation", in *Computer Graphics (ACM SIGGRAPH '88 Proceedings)*, vol. 22, pp. 75–84, (August 1988).

Appendix A: Concavity conjectures: case study of a differential area emitter

Let us consider the case of an infinite receiving plane and a single differential area for the emitter. In this case, due to the symmetries shared by the emitter and the receiver, there is only one parameter: the angle, called θ, between the normal of the emitter and a line parallel to the receiver, (see figure 16).

To express the position of a point on the receiver, we choose a set of axes related to the emitter: the first axes shares the direction of the projection of the normal of the emitter on the receiver, and the second axes is orthogonal to the first. The origin of our coordinate system is the projection of the emitting point. Using this set of coordinates, we have a simple expression for the point-to-area form-factor at any point $M(u, v)$ on the receiver (see figure 17 the aspect of the surface):

$$F(u,v) = \frac{dA}{\pi} \frac{u\cos(\theta) + \sin(\theta)}{(u^2+v^2+1)^2}$$

This value is only for $u\cos(\theta) + \sin(\theta) > 0$. If $u\cos(\theta) + \sin(\theta) \leq 0$, then of course $F(u, v) = 0$.

The C1 concavity conjecture

In this simple case, it is possible to explicitly compute the derivatives of the point-to-area form-factor. An explicit computation of the Hessian shows that it is definite if and only if the expression $S(u, v, \theta)$ is positive, where $S(u, v, \theta)$ is:

$$\begin{aligned} S \;=\; & -3u^4 - 8\tan\theta u^3 - 5\tan^2\theta u^2 - 4u^2v^2 + 3u^2 \\ & +4u\tan\theta - 8\tan\theta uv^2 \\ & -5\tan^2\theta v^2 - v^2 - v^4 + \tan^2\theta \end{aligned}$$

Although it is impossible to find an explicit solution of the equation $S(u, v, \theta) = 0$, it is possible to plot these solutions for different values of θ. Figure 18 shows the contour of the area where $S(u, v, \theta)$ is positive for different values of θ. Outside these areas, $S(u, v, \theta)$ is negative, and hence the Hessian matrix is indefinite. Inside these areas, S is positive, and the form-factor is concave.

An interesting point is the shape of the zones where the point-to-area form factor is concave. When $\theta = \frac{\pi}{2}$, it is of course a disc, due to the symmetries in the scene. When $\theta = 0$, it is a shape like a drop, that tapers to a point in $(0, 0)$. For intermediate values of θ, the zone has an intermediate shape between the drop and the disc, but this shape always appears to be convex.

The C2 concavity conjecture

If we now focus on the radiosity on a specific line $v = au + b$ on the receiving plane, we have, for the form-factor as a

function of u,

$$f(u) = \frac{dA}{\pi} \frac{u\cos(\theta) + \sin(\theta)}{(u^2 + (au+b)^2 + 1)^2}$$

The form-factor is equal to $f(u)$ if $u\cos(\theta)+\sin(\theta) > 0$. If $u\cos(\theta) + \sin(\theta) \leq 0$, then the form-factor is null.

It must be noted that $f(u)$ goes to zero when u goes to $\pm\infty$, and that $f(u)$ is equal to zero only for $u = u_0 = -\tan\theta$.

It is possible to compute the first and the second derivative of $f(u)$. The first derivative, $f'(u)$, is of the sign of a second degree polynomial in u, and the second derivative, $f''(u)$ is of the sign of a third degree polynomial in u. As a consequence, $f'(u)$ can change sign at most twice, and $f''(u)$ at most three times.

Since the function $f(u)$ goes to zero when u goes to $\pm\infty$, it must have one maximum between u_0 and $+\infty$, and one minimum between u_0 and $-\infty$. As a consequence, $f'(u)$ must change sign exactly twice. Let us call u_1 and u_2 the points where the first derivative changes sign ($u_1 < u_0 < u_2$).

$f'(u)$ also goes to zero when u goes to $\pm\infty$. As a consequence, it must have one minimum between u_2 and $+\infty$, and another between $-\infty$ and u_1, and it must have one maximum between u_1 and u_2. So the second derivative changes sign exactly three times. One of the point where the second derivative changes sign is smaller than u_1, which is smaller than u_0, and one of them is greater than u_2, which is greater than u_0.

Then the second derivative changes sign at least once and at most twice on $[u_0, +\infty]$. When u goes to $+\infty$, f is convex, and f'' is positive. So we just proved that f'' can be negative only over a unique bounded segment on $[u_0, +\infty]$.

The form-factor on the line is equal to $f(u)$ for $u > u_0$, and null everywhere else. So the form-factor on a line is concave only over a unique bounded segment. This proves the C2 conjecture for a differential area emitter.

Figure 19 shows an example of such a $f(u)$ function, along with its first and second derivatives. It can be noted that this function is concave over a single segment, and convex everywhere else.

Appendix B: Effective computation of the form-factor derivatives

In this section, we show how it is possible to compute the derivatives of the point-to-area form-factor with little additional computation expense.

In particular, it is shown that the computation of the exact value of the form-factor derivatives is always cheaper than the computation of an approximate value using several form-factor samples. For example, the cost of computing the form-factor gradient is 30 %, while computing an approximate value of the gradient would require two form-factor samples, thus increasing computation time by 100 %

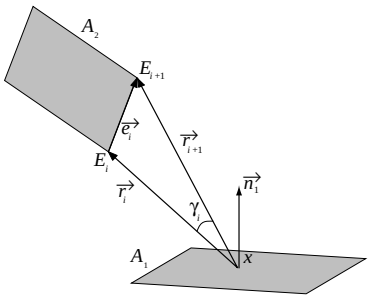


Figure 20: *Notation when the emitter is a polygon.*

$F = 0$
foreach edge $[E_i E_{i+1}]$
$\vec{r_i} = E_i - x$
$\vec{r}_{i+1} = E_{i+1} - x$
$\text{crossprod} = \vec{r_i} \times \vec{r}_{i+1}$
$gamma = \arccos\left(\frac{\vec{r_i}\cdot\vec{r}_{i+1}}{r_i r_{i+1}}\right)$
$I_1 = \frac{gamma}{\|\text{crossprod}\|}$
$\text{mixt} = \vec{n_1} \cdot \text{crossprod}$
$F- = I_1 \text{mixt}$
$F* = \frac{1}{2\pi}$

Figure 21: *Pseudo-Code for computing the form-factor.*

The Point-to-Area Form-Factor

Let us recall that the point-to-area form-factor from a point x on a patch A_1 to a patch A_2 (see figure 1) can be expressed as a contour integral:

$$F(x) = -\vec{n_1} \cdot \frac{1}{2\pi} \oint_{\partial A_2} \frac{\vec{r}_{12} \times d\vec{\ell_2}}{\|\vec{r}_{12}\|^2}$$

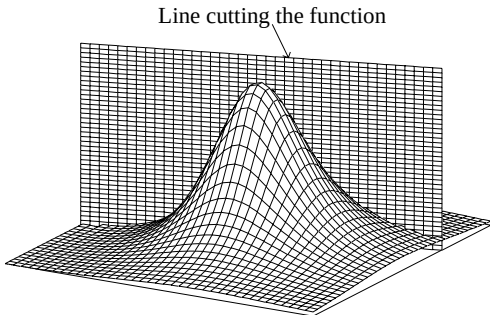
For the explicit derivation of this contour integral from the equation 2, see Siegel and Howell[14].

In the case where the emitter is a polygon, this expression simplifies to a finite sum:

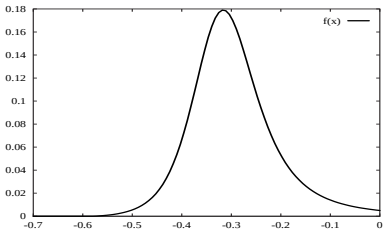
$$F(x) = \frac{1}{2\pi} \vec{n_1} \cdot \sum_i \vec{\gamma_i} \tag{6}$$

where $\vec{\gamma_i}$ is the vector of norm γ_i, and of direction the cross-product $\vec{r_i} \times \vec{r}_{i+1}$ (see figure 20).

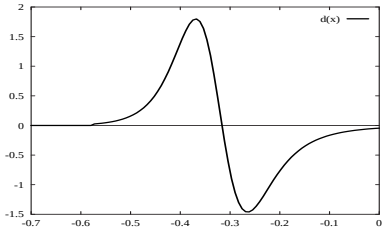
An example pseudo-code for computing the form-factor using equation 6 can be found in figure 21. This pseudo-code makes use of the standard 3D operations like addition, cross-product and dot product.



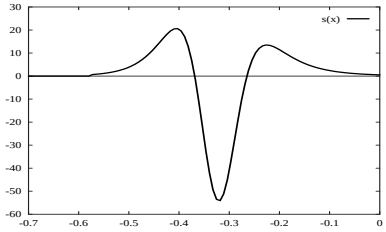
A line cutting through the radiosity function



The radiosity function on the line



The first derivative of the radiosity function on the line



The second derivative is negative only over a segment.

Figure 19: *The radiosity on any line on the receiving plane is concave only over a segment.*

Form-Factor Gradient

The point-to-area form-factor gradient can be easily computed by derivation of the previous formula (see Arvo[15], or Holzschuch[17]):

$$\nabla F(x) = -\frac{-1}{2\pi}\sum_i \vec{n}_1 \times \vec{e}_i I_1 + 2\vec{n}_1 \cdot (\vec{r}_i \times \vec{r}_{i+1})\,(\vec{r}_i I_2 + \vec{e}_i J_2)$$

With:

$$I_1 = \frac{\gamma_i}{\|\vec{e}_i \times \vec{r}_i\|}$$

$$I_2 = \frac{1}{2\|e_i \times \vec{r}_i\|^2}\left(\frac{\vec{e}_i \cdot \vec{r}_{i+1}}{r_{i+1}^2} - \frac{\vec{e}_i \cdot \vec{r}_i}{r_i^2} + e_i^2 I_1\right)$$

$$J_2 = \frac{1}{2e_i^2}\left(\frac{1}{r_i^2} - \frac{1}{r_{i+1}^2}\right) - \frac{\vec{e}_i \cdot \vec{r}_i}{e_i^2} I_2$$

The code in figure 21 for computing the form-factor can be extended for computing the gradient. Figure 22 shows the extension of the pseudo-code needed for computing simultaneously the point-to-area form-factor and its gradient (we did not include the part of the code that is exactly identical). As can be seen, most of the costly computations like inverse trigonometric functions have been done for the form-factor, and do not need to be redone for the gradient.

The exact extra cost of computing the gradient depends on the computer and on the compiler used. On an R4000 SGI with the standard cc compiler, it is 30 % (see Holzschuch[16, 17]).

What is fundamental is that it actually costs much less to compute the exact value for the gradient than it would cost to compute two radiosity values, and then to approximate the gradient using these values.

$F = 0$
$\vec{G} = \vec{0}$
foreach edge $[E_i E_{i+1}]$
  .
  .
  .
  $F- = I_1 \mathrm{mixt}$
  $\vec{e}_i = \overrightarrow{E_i E_{i+1}}$
  $I_2 = \frac{\vec{e}_i \cdot \vec{r}_{i+1}}{r_{i+1}^2} - \frac{\vec{e}_i \cdot \vec{r}_i}{r_i^2} + e_i^2 I_1$
  $I_2 / = 2\mathrm{crossprod}^2$
  $J_2 = 0.5\left(\frac{1}{r_i^2} - \frac{1}{r_{i+1}^2}\right) - \vec{e}_i \cdot \vec{r}_i I_2$
  $J_2 / = e_i^2$
  $\vec{G} + = (\vec{n}_1 \times \vec{e}_i) I_1 + 2\mathrm{mixt}(\vec{r}_i I_2 + \vec{e}_i J_2)$
$F* = \frac{1}{2\pi}$
$\vec{G}* = -\frac{1}{2\pi}$

Figure 22: *Pseudo-Code for computing the gradient of the form-factor.*

```
F = 0
G⃗ = 0⃗
H = 0
foreach edge [E_i E_{i+1}]
    .
    .
    .
    G⃗ += (n⃗_1 × e⃗_i) I_1 + 2mixt(r⃗_i I_2 + e⃗_i J_2)
    I_3 = (e⃗_i · r⃗_{i+1}) / r⁴_{i+1} − (e⃗_i · r⃗_i) / r⁴_i + e_i² I_2
    I_3 /= 4crossprod²
    J_3 = 0.25 ( 1/r⁴_i − 1/r⁴_{i+1} ) − e⃗_i · r⃗_i I_3
    J_3 /= e_i²
    K_3 = I_2 − r_i² I_3 − 2 e⃗_i · r⃗_i J_3
    K_3 /= e_i²
    H += −mixt I_2 I + Q(r⃗_i I_2 + e⃗_i J_2, n⃗_1 × e⃗_i)
    +2mixt(Q(r⃗_i, r⃗_i) I_3 + Q(e⃗_i, e⃗_i) K_3 + 2 J_3 Q(e⃗_i, r⃗_i))
F* = 1/(2π)
G⃗* = −1/(2π)
H* = 1/π
```

Figure 23: *Pseudo-Code for computing the first two derivatives of the form-factor.*

Hessian matrix for the point-to-area form-factor

The point-to-area form factor Hessian matrix can also be computed by derivation of Equation 6 (see Holzschuch[17]):

$$\begin{aligned} H \;=\; & -\frac{1}{\pi}\sum_i Q\left(\vec{n}_1 \times \vec{e}_i, \vec{r}_i I_2 + \vec{e}_i J_2\right) \\ & -\vec{n}_1 \cdot (\vec{r}_i \times \vec{e}_i) I_2 I \\ & +2\vec{n}_1 \cdot (\vec{r}_i \times \vec{e}_i)(Q(\vec{r}_i, \vec{r}_i) I_3 \\ & +Q(\vec{e}_i, \vec{e}_i) K_3 + 2 J_3 Q(\vec{r}_i, \vec{e}_i)) \end{aligned}$$

We use the following notation:

$$\begin{aligned} Q(\vec{a}, \vec{b}) \;&=\; \vec{a}^{\mathrm{t}}\vec{b} + \vec{b}^{\mathrm{t}}\vec{a} \\ I_3 \;&=\; \frac{1}{4}\frac{1}{\|\vec{e}_i \times \vec{r}_i\|^2}\left(\frac{\vec{e}_i \cdot \vec{r}_{i+1}}{r_{i+1}^4} - \frac{\vec{e}_i \cdot \vec{r}_i}{r_i^4} + 3e_i^2 I_2\right) \\ J_3 \;&=\; \frac{1}{4e_i^2}\left(\frac{1}{r_i^4} - \frac{1}{r_{i+1}^4}\right) - \frac{\vec{r}_i \cdot \vec{e}_i}{e_i^2} I_3 \\ K_3 \;&=\; \frac{1}{e_i^2}\left(I_2 - r_i^2 I_3 - 2(\vec{r}_i \cdot \vec{e}_i) J_3\right) \end{aligned}$$

The code for computing the form-factor and the gradient can be extended to compute the second derivative as well. Figure 23 shows the extension of the pseudo-code needed for computing simultaneously the point-to-area form-factor and its first two derivatives (we did not include the part of the code that is exactly identical). Once again, recycling geometric computations previously done reduces the cost of computing the Hessian matrix, even if the cost is still high since matrix operations are quite expensive: a single matrix addition has the same cost as 9 standard additions.

The exact extra cost of computing the Hessian matrix depends on the computer and on the compiler. On a R4000 SGI, with the standard cc compiler, it is 80 % of the cost of the form-factor alone (see Holzschuch[17]), meaning that the overall cost of computing the point-to-area form-factor and its first two derivatives is 2.1 times the cost of computing the form-factor alone. Notice it is much faster to compute the exact value than it would be to compute an approximate Hessian matrix — which would require seven separate form-factor computations.

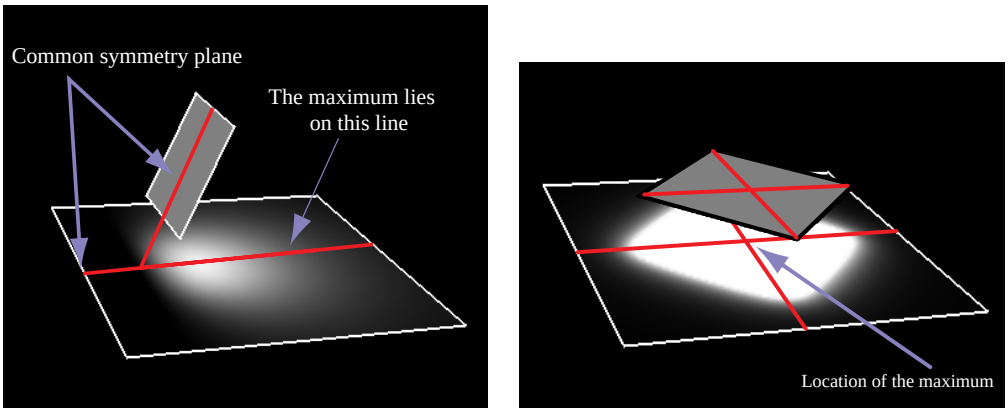


Figure 24: *The symmetries of the scene can help find the location of the maximum.*

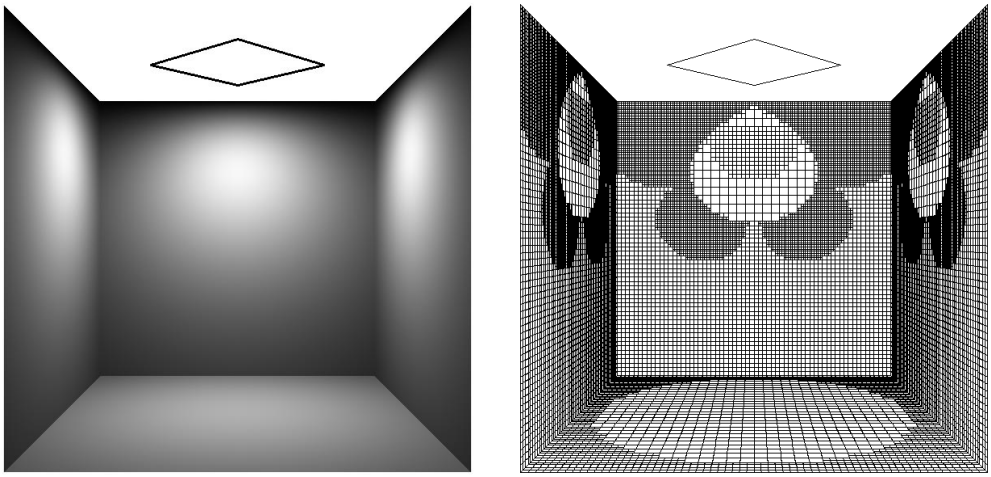


Figure 25: *Direct illumination with our refinement criterion, unoccluded scene.*

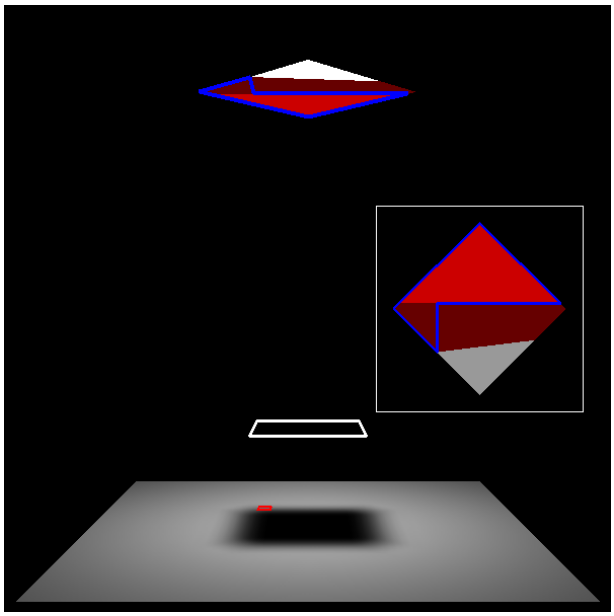


Figure 26: *Minimal and maximal emitter for a simple configuration.*

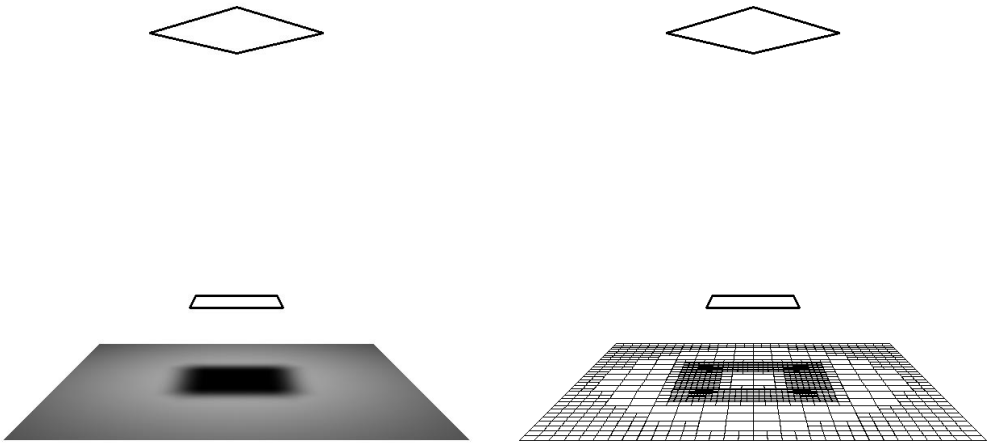


Figure 27: *Direct illumination with our refinement criterion, with one occluder.*

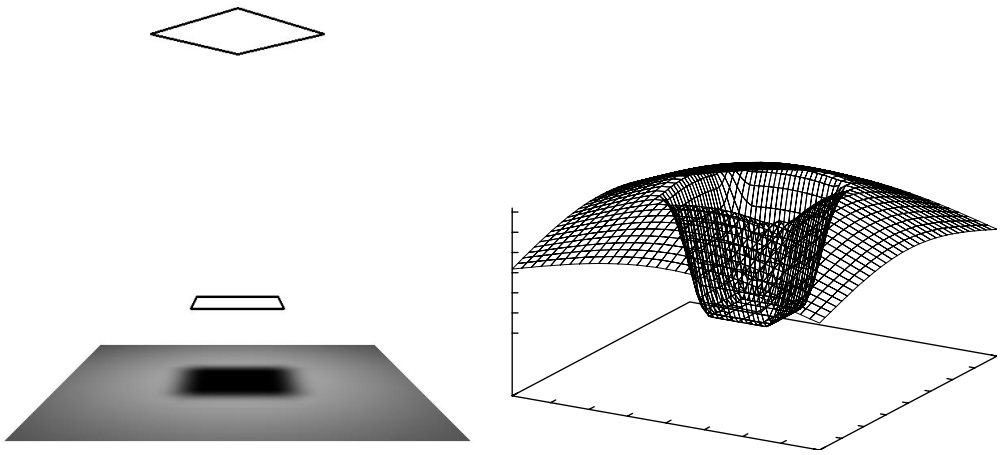


Figure 28: *With generic occluders, the unimodality conjectures do not hold.*

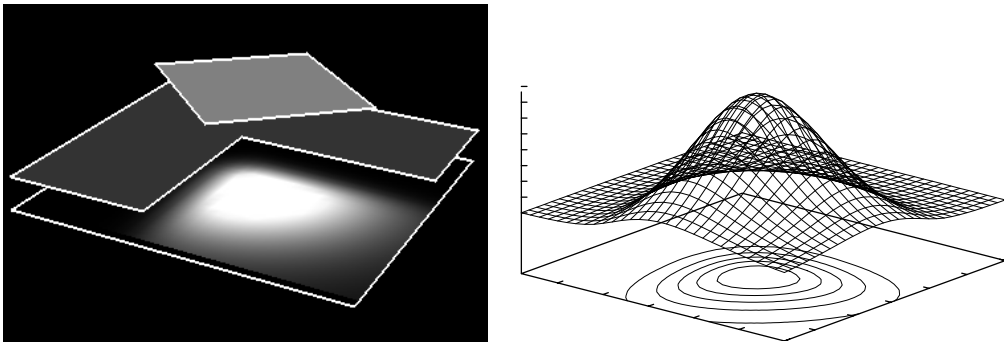


Figure 29: *With certain occluders, the unimodality conjectures still holds.*