

# A Progressive Importance-Driven Rendering Algorithm

Philippe Bekaert and Yves D. Willems  
Katholieke Universiteit Leuven, Department of Computer Science  
Celestijnenlaan 200 A, B-3001 Leuven, Belgium  
e-mail: Philippe.Bekaert@cs.kuleuven.ac.be

## Abstract

This paper describes a progressive and importance-driven discrete three-point transport algorithm for solving the global illumination problem. A new use of importance enables us to develop an algorithm in which as much progress as possible is made in early stages of the solution process unlike existing importance-driven finite-element methods.

## 1 Introduction and related work

The global illumination problem is the problem to be solved when we want to visualise a scene (e.g. a building being designed by an architect) taking into account not only the direct lighting from light sources but also all possible interreflections between objects in the scene. The problem is described by an integral equation usually called the *rendering equation* [9]. Solving the global illumination problem requires solving this (non-trivial) integral equation. Two classes of methods are currently used achieve this:

A first class of methods is based on *Monte-Carlo* techniques [9, 11] such as stochastic raytracing. They are elegant and can be used for complex geometries and reflection functions, but are known to converge slowly. Current research focuses on techniques to speed up convergence.

A second class uses a *finite-element* approach [5]. The *radiosity* method [3, 4, 6] is a finite-element method that solves the rendering equation for surfaces that exhibit only diffuse (Lambertian) reflection. The main problem with finite-element techniques in general, where specular and glossy reflections have to be taken into account also, is the huge amount of storage and time needed for complex environments [8, 12].

*Hierarchical methods* [7] help to overcome this problem by reducing the number of formfactors to be computed to achieve a given accuracy: patches and elements are subdivided into smaller elements, refining the *interactions* between them, until the estimated error on the formfactor becomes smaller than a predefined error bound (*F-refinement*). A further reduction of complexity is achieved by also taking into account how much light is transported in a to be refined interaction, leading to so-called *BF-refinement*. The idea is that interactions between patches carrying only a small amount of radiosity should not be refined as much as interactions carrying a large amount of radiosity. This method can also be extended to glossy reflection [1].

Both classes can be further subdivided in shooting and gathering methods: *Gathering* algorithms are described by the traditional rendering equation [9]. Stochastic raytracing is a well-known gathering method: in order to compute the flux of light being emitted towards an observer in a scene, rays are traced from the observer position to the objects in the scene. For determining

the “color” of a ray, reflection rays are spawned over the entire (hemi)sphere of outgoing directions from the nearest intersection point. The “colors” of these rays are determined recursively and combined to be the color of the observer-ray. What actually happens is that the light coming in from a set of (hemi)spherical directions is gathered and reflected towards the observer. *Gauss-Seidel radiosity* is another well-known gathering method.

*Progressive radiosity* [3] is an example of a *shooting* method. The idea behind progressive radiosity is to shoot light from the light sources to all other patches of a scene and to reflect it in the environment. One therefore keeps per patch not only the total radiosity, but also the “unshot” radiosity. The unshot radiosity is the radiosity a patch receives from other patches in the environment which has not been reflected yet. By repeatedly choosing the patch with most unshot radiosity and shooting its unshot radiosity to all other patches, rough approximations for the resulting image are obtained quite fast, which are then gradually improved: progressive radiosity provides fast feedback by making as much progress as possible in the beginning of the solution process.

The adjoint of the rendering equation, the so-called *potential equation* [10], describes the global illumination problem from another point of view: instead of computing the total *radiance* that is emitted towards an observer from the visible points in the scene, one tries to compute the *importance*, also called *visual potential*, received by the points on the light sources from all directions. The importance a point receives from a given direction tells us what fraction of the light emitted by that point along that direction contributes to the final image. Visibility from the observer position can be thought of as directly received or *initial importance*: the light emitted by a visible point, along the direction from that point to the observer, fully contributes to the flux of light reaching the observer. If the point were not visible, it would only contribute indirectly, through reflections to and by visible surfaces. The potential equation describes how importance is propagated in an environment. This is explained in more detail in §2.

The use of importance in finite-element solutions to the rendering equation has not been investigated until recently. Importance is used in [13] to control the hierarchical subdivision of patches, in addition to the estimated error on the formfactor and the radiosity transported between two patches or elements. The idea is that interactions that contribute not so much to a particular view of the scene one wants to generate, should be less refined than interactions that are more “important”. An extension to glossy reflection is presented in [2].

A disadvantage of the methods presented in [2, 13] is that they do not provide feedback as soon as possible. In these methods radiance is computed by gathering. In this paper we will present an importance-driven method in which radiance is determined by shooting. We will not use importance to control the refinement of interactions and the subdivision of patches, but will use it when choosing the next most contributing patch or interaction in a better progressive algorithm.

§3 describes how our algorithm is obtained from the equations in §2. The extension to an importance-driven algorithm is explained in §4. Some preliminary results are presented in §5.

## 2 Discrete Three Point Transport

Suppose surfaces in a scene have been subdivided into patches. Let’s also assume that emitted radiance, received importance and reflection characteristics are constant over these patches<sup>1</sup> In order to render an image, we need for each patch  $i$  the average radiance emitted towards the

---

<sup>1</sup>This is the basic assumption in finite-element methods with constant basis functions, see e.g. [5].

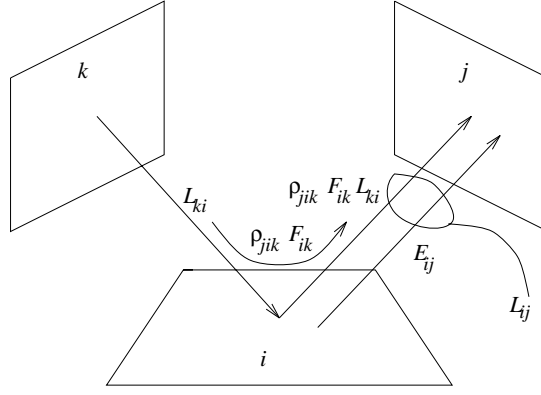


Figure 1: propagation of radiance

observer by the points on that patch. However, we will base our discussion on the total flux of light  $\Phi [W]$  towards the observer since the duality of the radiance- and importance-based equations can be expressed easier this way. This does not introduce any problems since the average radiance can be derived from the flux. In absence of participating media, this flux depends only on the geometry and the reflection properties of the surfaces in the scene. One expression for the flux, based on the rendering equation [9], is:

$$\Phi = \sum_i \sum_j L_{ij} A_i F_{ij} N_{ij} \quad (1)$$

with [1]

$$L_{ij} = E_{ij} + \sum_k \rho_{jik} F_{ik} L_{ki} \quad (2)$$

where

- $L_{ij}$  is the average radiance  $[W/m^2sr]$  emitted from points on patch  $i$  to points on patch  $j$ ;
- $A_i$  is the surface area  $[m^2]$  of patch  $i$ ;
- $F_{ij}$   $[sr]$  is the  $\pi$  times the formfactor from patch  $i$  to patch  $j$ ;
- $N_{ij}$  is the initial importance (dimensionless) points on patch  $i$  receive from points on patch  $j$  (see below);
- $E_{ij}$  is the self-emitted radiance  $[W/m^2sr]$  emitted from points on patch  $i$  to points on patch  $j$ . The self-emitted radiance is non-zero only for patches on light sources;
- $\rho_{jik}$  is the average value of the Bidirectional Reflection Distribution Function (*BRDF*)  $f_r(x_j, x_i, x_k)$   $[1/sr]$  for reflection at points  $x_i$  on patch  $i$  of light coming in from the direction to points  $x_k$  on patch  $k$  along the direction to points  $x_j$  on patch  $j$ .

The initial potential  $N_{ij}$  requires some more explanation: We have treated the observer as a patch of differential area and normal equal to the viewing direction. The initial potential  $N_{ij}$  is nonzero only if  $j$  indicates that observer “patch”. In that case we define

$$N_{ij} = \frac{1}{A_i} \int_{A_i} d\mu_x \frac{\cos \theta_{eye}}{r_{x,eye}^2} \text{vis}(x, \text{eye}) \quad (3)$$

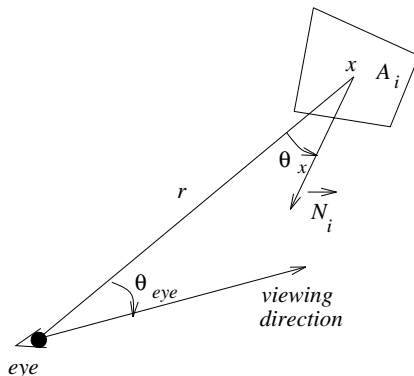


Figure 2: Geometry for computing the initial potential

and

$$A_i F_{ij} N_{ij} = \int_{A_i} d\mu_x \frac{\cos \theta_x \cos \theta_{eye}}{r_{x,eye}^2} \text{vis}(x, \text{eye}) \quad (4)$$

where (see figure 2)

- $d\mu_x$  is a differential area around point  $x$  on patch  $i$ ;
- $\theta_x$  is the angle between the direction from  $x$  to the observer position and the normal on patch  $i$  in point  $x$ ;
- $\theta_{eye}$  is the angle between the direction from the observer position to the point  $x$  and the viewing direction;
- $r_{x,eye}$  is the distance between point  $x$  and the observer position;
- $\text{vis}(x, \text{eye})$  is 1 if point  $x$  is visible from the observer position and 0 if not.

The terms in the sum in the right part of equation (1) are zero unless patch  $j$  is the observer “patch”. The double sum thus reduces to a single sum over all patches visible from the viewing position. It is the radiance  $L_{ij}$  of these patches  $i$  towards the observer that is used for creating an image.

Equations (1) and (2) have a clear physical interpretation: (1) describes how the flux of light towards the observer is the sum of the total light emitted towards the observer by all patches  $i$  in the scene, weighted by the initial importance received by those patches. Equation (2) (see figure 1) describes how the total radiance  $L_{ij}$  emitted from patch  $i$  to patch  $j$  is the sum of the self-emitted radiance  $E_{ij}$  from  $i$  to  $j$  (if patch  $i$  is on a light source) and the radiances  $\rho_{jik} F_{ik} L_{ki}$  emitted by each third patch  $k$  reflected by patch  $i$  towards  $j$ .  $\rho_{jik} F_{ik}$  is commonly called the *area reflectance* of patch  $i$  for light coming in from patch  $k$  being reflected to patch  $j$ : it is the ratio of light emitted by  $k$  and reflected by  $i$  towards  $j$  over the total light coming in from  $k$ . It is interesting to see how in the case of diffuse (Lambertian reflection), when  $L_{ij}$  and  $\rho_{jik}$  depend only on the patch  $i$ , this equation reduces to the classical radiosity equation ( $L_{ij}$  is substituted by  $L_i$  and  $\rho_{jik}$  by  $\rho_i$ ):

$$L_i = E_i + \rho_i \sum_k F_{ik} L_k \quad (5)$$

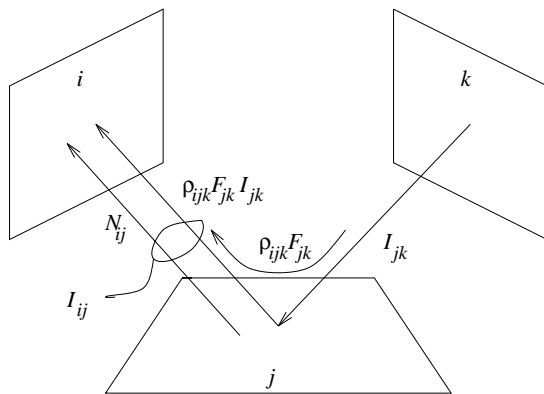


Figure 3: propagation of importance

Another expression for the total flux of light to the observer is based on the potential equation [10]:

$$\Phi = \sum_i \sum_j E_{ij} A_i F_{ij} I_{ij} \quad (6)$$

with<sup>2</sup>

$$I_{ij} = N_{ij} + \sum_k \rho_{ijk} F_{jk} I_{jk} \quad (7)$$

where  $I_{ij}$  is the total importance patch  $i$  receives from patch  $j$  (see figure 3).

These equations also have a clear physical interpretation: equation (6) says that the flux of light towards the observer is the sum of the self-emitted light of patches  $i$  on light sources towards all other patches  $j$ , weighted by the total importance  $i$  receives from  $j$ . Equation (7) (see figure 3) describes how importance is propagated in the scene: the importance  $I_{ij}$ , received by patch  $i$  from patch  $j$  is the sum of the initial importance  $N_{ij}$  (which is nonzero only if  $j$  indicates the observer “patch”) and the importances  $\rho_{ijk} F_{jk} I_{jk}$  received by  $j$  from each third patch  $k$ , reflected towards  $i$ .

The equations in this section describe what is called *discrete three-point transport*. It is explained in the next section how a progressive algorithm can be obtained from these equations.

### 3 A Progressive Discrete Three-Point Transport Algorithm

We need to compute the flux of light emitted by the scene to the viewing position. One expression for this flux is given by equation (1) where the total radiance  $L_{ij}$  transported in each *interaction*  $i \rightarrow j$  between patches  $i$  and  $j$  is given by the infinite sum of equation (2).

Cohen et al. [3] have developed a shooting method for computing the radiances  $L_{ij}$  for the case where all surfaces in the scene exhibit diffuse reflection only (equation (5)). This method can easily be adapted for the general case (equation (2)). The result is algorithm 1.

---

<sup>2</sup>A derivation of equation 7 can be found by constructing the adjoint of equation (2) [2], but can also be obtained directly from the potential equation [10] with the techniques outlined in [5].

### Algorithm 1 *Basic Shooting Algorithm*

1. Initialize  $L_{ij} = \Delta L_{ij} \leftarrow E_{ij}$  for all patches  $i$  and  $j$ ;
2. repeat until convergence is reached
  - (a) for each interaction  $k \rightarrow i$  do
    - i. for each patch  $j$  do
      - A.  $\Delta \text{rad} \leftarrow \Delta L_{ki} A_k F_{ki} \rho_{kij}$ ;
      - B.  $A_i L_{ij} \leftarrow A_i L_{ij} + \Delta \text{rad}$ ;
      - C.  $A_i \Delta L_{ij} \leftarrow A_i \Delta L_{ij} + \Delta \text{rad}$ .
    - ii.  $\Delta L_{ki} \leftarrow 0$

However, a better progressive algorithm is obtained by solving in sorted order. This is accomplished by not iterating over all interactions  $k \rightarrow i$  in step 2a, but by choosing only the “next most contributing” interaction. This requires that we have a criterion for deciding which is the “next most contributing” interaction however. One such criterion is the amount  $\Delta L_{ki} A_k F_{ki}$  of unshot radiance to be propagated in an interaction  $k \rightarrow i$ , leading to algorithm 2.

### Algorithm 2 *Progressive Algorithm*

1. Initialize  $L_{ij} = \Delta L_{ij} \leftarrow E_{ij}$ ;
2. Repeat until convergence is reached
  - (a) choose the interaction  $k \rightarrow i$  for which  $\Delta L_{ki} A_k F_{ki}$  is largest;
  - (b) for each patch  $j$  do
    - i.  $\Delta \text{rad} \leftarrow \Delta L_{ki} A_k F_{ki} \rho_{kij}$ ;
    - ii.  $A_i L_{ij} \leftarrow A_i L_{ij} + \Delta \text{rad}$ ;
    - iii.  $A_i \Delta L_{ij} \leftarrow A_i \Delta L_{ij} + \Delta \text{rad}$ .
  - (c)  $\Delta L_{ki} \leftarrow 0$ .

For fast feedback, the radiances  $L_{ij}$  after each step should be used to render an image e.g. by graphics hardware. Algorithm 2 then yields a rough image very quickly, which gracefully converges to a realistic image. The algorithm is extended to an importance-driven algorithm in the next section.

## 4 An Importance-Driven Progressive Algorithm

It is also possible to use equations (6) and (7) to compute the flux of light reaching the observer of a scene. With this pair of equations, the flux is obtained by computing the importances  $I_{ij}$ . The importances  $I_{ij}$  can be computed with algorithm 3 which is analogous to algorithm 2 for computing the radiances  $L_{ij}$ .

### Algorithm 3 Computing Importance

1. Initialize  $I_{ij} = \Delta I_{ij} \leftarrow N_{ij}$ ;
2. Repeat
  - (a) choose the interaction  $i \rightarrow j$  for which  $\Delta I_{ij} F_{ij}$  is largest;
  - (b) for each patch  $k$  do
    - i.  $\Delta \text{imp} \leftarrow \Delta I_{ij} F_{ij} \rho_{kij}$ ;
    - ii.  $I_{ki} \leftarrow I_{ki} + \Delta \text{imp}$ ;
    - iii.  $\Delta I_{ki} \leftarrow \Delta I_{ki} + \Delta \text{imp}$ .
  - (c)  $\Delta I_{ij} \leftarrow 0$ .

Unfortunately, we need the radiance emitted by each (visible) patch towards the observer. So, this algorithm is not directly suited for rendering an image. However, the importance  $I_{ij}$  tells us what fraction of the radiance carried in the interaction  $i \rightarrow j$  contributes to a particular view one wants to generate. Smits [13] reported that the use of importance can decrease the time needed to render an image for one or a few particular viewing positions by several orders of magnitude.

In [13] radiance was computed by gathering and importance by shooting. Also computing radiance by shooting yields a better progressive algorithm. In that case, the computed importances  $I_{ij}$  should be taken into account when searching for the “next most contributing” interaction in step 2(b)i) of algorithm 4. This is accomplished by looking for the interaction for which the product

$$\Delta L_{ki} A_k F_{ki} I_{ki} \tag{8}$$

is largest instead of just the interaction from which most unshot radiance can be propagated.

It is clear that the criterion used to choose the “next most contributing” interaction does not affect the radiance solution, but only the order of computations. This has the advantage that, like the method in [13], only the importance computations need to be done over when changing viewpoint, changing the order of computations for fast feedback for that new viewpoint. The disadvantage of using shooting for both radiance and importance is that the radiance- and importance computations are more independent than in [2, 13], where radiance is gathered and importance shot simultaneously. This should not be too much a problem however since far most of the time is spent in the computing the formfactors, which are the same for propagating importance or radiance and are kept in a data structure in an implementation.

## 5 Some Results

The figures at the end of this article show some results obtained with algorithm 4 for a diffuse environment.

The first thing our implementation does is the computation of the direct lighting  $E_{ij}$  terms. While computing these terms, patches (and their interactions) are subdivided to allow an accurate reproduction of shadow boundaries. Plate 1a. shows the result of this initial phase for a labyrinth scene containing 378 initial patches, which were subdivided into about 7000 elements (14500 interactions). This initial phase took about 2 minutes on our HP 9000/715-50 workstations.

**Algorithm 4** *Importance-Driven Progressive Algorithm*

1. Initialize  $L_{ij} = \Delta L_{ij} \leftarrow E_{ij}$  and  $I_{ij} = \Delta I_{ij} \leftarrow N_{ij}$ ;
2. Repeat until convergence is reached
  - (a) *Propagate importance*
    - i. choose the interaction  $i \rightarrow j$  for which  $\Delta I_{ij} F_{ij}$  is largest;
    - ii. for each patch  $k$  do
      - A.  $\Delta \text{imp} \leftarrow \Delta I_{ij} F_{ij} \rho_{kij}$ ;
      - B.  $I_{ki} \leftarrow I_{ki} + \Delta \text{imp}$ ;
      - C.  $\Delta I_{ki} \leftarrow \Delta I_{ki} + \Delta \text{imp}$ .
    - iii.  $\Delta I_{ij} \leftarrow 0$ .
  - (b) *Propagate radiance*
    - i. choose the interaction  $k \rightarrow i$  for which  $\Delta L_{ki} A_k F_{ki} I_{ki}$  is largest;
    - ii. for each patch  $j$  do
      - A.  $\Delta \text{rad} \leftarrow \Delta L_{ki} A_k F_{ki} \rho_{kij}$ ;
      - B.  $A_i L_{ij} \leftarrow A_i L_{ij} + \Delta \text{rad}$ ;
      - C.  $A_i \Delta L_{ij} \leftarrow A_i \Delta L_{ij} + \Delta \text{rad}$ .
    - iii.  $\Delta L_{ki} \leftarrow 0$ .

Once the initial interactions have been set up, the user has the choice to continue solving the global illumination problem with or without the use of importance. Plates 1b. and 1c. show the results obtained without using importance (algorithm 2) after resp. 60 and 250 iterations taking resp. about 6 and 30 minutes: One clearly observes how the initial solution, showing only direct lighting, is gradually improved. Solving without importance has the best global convergence: one looks for a solution which is of equal accuracy over the scene allowing to walk through it everywhere without need for further expensive calculations. Although this might be an advantage in some situations, the time spent on such a solution is mostly a disadvantage.

One often wants to render a scene quickly for one particular viewing position, still being able to move to another position without having to repeat all computations. The use of importance will allow us to do so. Figure 2a. shows the initial solution for one room in the labyrinth. Plate 2b. is the result after computing the initial importances  $N_{ij}$  for this view (taking less than one second using graphics hardware), propagating the  $N_{ij}$  terms once (so the eye point-patch has no unshot importance, taking about 10 seconds) and then alternating steps for propagating radiance and importance (algorithm 4). The view hardly changed after one minute. Plate 2b. is the result after 60 iterations (2 minutes). Plate 3a. was obtained after changing the viewpoint. Recalculating importance and then alternating 50 steps for distributing radiance and importance yielded 3b. (3 minutes). It took more than 40 minutes to obtain images comparable to plate 2b. and 3b. without using importance.

Plates 2c. and 3c. show how only interactions that are relevant for a particular view are treated when using importance. Compare with plate 1d. which shows the interactions used to obtain plate 1b. where no importance was used.

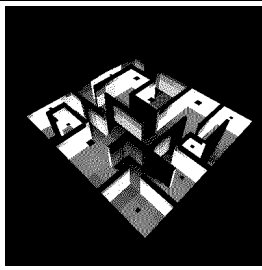
## 6 Conclusion

Existing importance-driven finite-element methods for solving the global illumination problem do not make as much progress as possible in the early stages of the solution phase. A different use of importance, in a method where both radiance and importance are computed by shooting, yields a better progressive and importance-driven algorithm. We have used importance to control the order of computations whereas in existing methods it is used to control the subdivision of patches in [13] or the refinement of interactions [2].

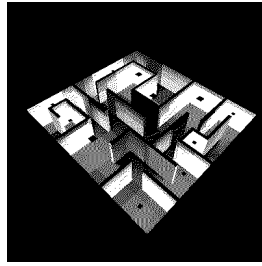
It is possible to combine our use of importance with the use of importance as in [13, 2]. A combination of both is likely to yield even larger performance increases.

## References

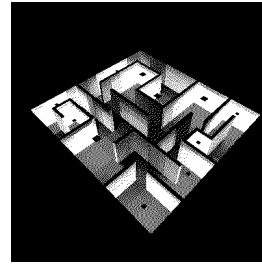
- [1] L. Aupperle and P. Hanrahan. A hierarchical illumination algorithm for surfaces with glossy reflection. In *Computer Graphics Proceedings*, pages 155–162, 1993.
- [2] L. Aupperle and P. Hanrahan. Importance and discrete three-point transport. In *Proceedings of the fourth Eurographics Workshop on Rendering, Paris, France*, pages 85–94, 1993.
- [3] M. F. Cohen, S. E. Chen, J. R. Wallace, and D. P. Greenberg. A progressive refinement approach to fast radiosity image generation. *Computer Graphics*, 22(4):75–84, 1988.
- [4] M. F. Cohen, D. P. Greenberg, D. S. Immel, and P. J. Brock. An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, 6(3):25–35, 1986.
- [5] M. F. Cohen and J. R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, 1993.
- [6] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Bataille. Modeling the interaction of light between diffuse surfaces. *Computer Graphics*, 18(3):213–222, 1984.
- [7] P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics*, 25(4):197–206, 1991.
- [8] D. S. Immel, M. F. Cohen, and D. P. Greenberg. A radiosity method for non-diffuse environments. *Computer Graphics*, 20(4):133–142, 1986.
- [9] J. T. Kajiya. The rendering equation. *Computer Graphics*, 20(4):143–150, 1986.
- [10] S. Pattanaik and S. Mudur. The potential equation and importance in illumination computations. *Computer Graphics Forum*, 12:131–136, 1993.
- [11] P. Shirley and C. Wang. Distribution ray tracing: Theory and practice. In *Proceedings of the Third Eurographics Workshop on Rendering, Bristol, UK*, pages 33–43, 1992.
- [12] F. X. Sillion, J. R. Arvo, S. H. Westin, and D. P. Greenberg. A global illumination solution for general reflectance distributions. *Computer Graphics*, 25(4):187–196, 1991.
- [13] B. E. Smits, J. R. Arvo, and D. H. Salesin. An importance-driven radiosity algorithm. *Computer Graphics*, 26(2):273–282, 1992.



1a.



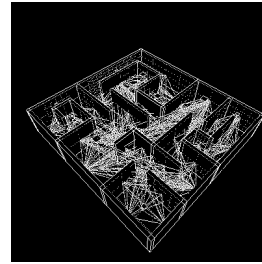
1b.



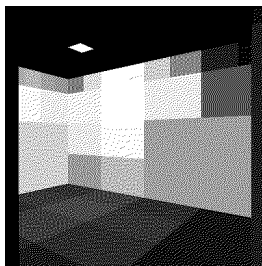
1c.

**Importance-Driven Progressive Discrete Three-Point Transport:**

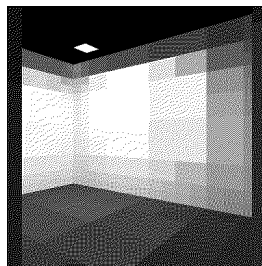
The figures on this page show the results of this method for a labyrinth scene: Figures 1a to 1c were produced without making use of importance: 1a is the result of an initial phase during which direct lighting is calculated (2 minutes), 1b was obtained after 60 iterations (6 minutes) and 1c after 250 iterations (30 minutes). One clearly observes the progressiveness of the method. Figures 2 and 3 illustrate the use of importance for producing fast solutions for particular views: 2a is the result of the initial phase, evolving into 2b after less than 2 minutes more. 3a is a view obtained next by changing the viewpoint. Recalculating importance and letting the algorithm go for another 3 minutes yields 3b. 1d, 2c and 3c show the interactions calculated to obtain 1b, 2b and 3b. 2c and 3c show clearly how by use of importance only relevant interactions are treated.



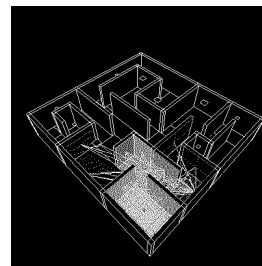
1d.



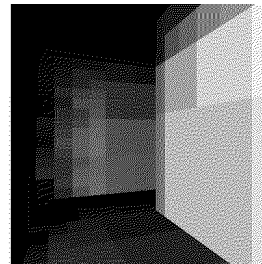
2a.



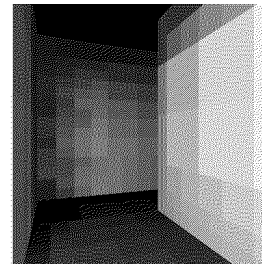
2b.



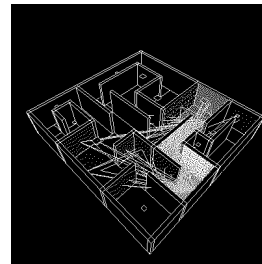
2c.



3a.



3b.



3c.