# A Two-Pass Physics-Based Global Lighting Model

Kadi Bouatouch Pierre Tellier\*

IRISA, Campus de Beaulieu 35042 Rennes Cedex, FRANCE Tel: +33 99.84.72.58, Fax: +33 99.38.38.32 kadi@irisa.fr, tellier@irisa.fr

#### Abstract

This paper describes a two-pass implementation of a physics-based global lighting model. This latter uses a physics-based reflection model, spectral distribution of light powers, and does not make any assumption on the specular behavior of materials. The scenes are discretized into points instead of patches. Hence, any kind of surface can be used without having to break down it into small planar patches. A data structure, named visibility graph, is built to efficiently evaluate the visibility between the sample points of the scene. Even though the photometric properties of surfaces (reflection, transmission, roughness, emitted powers...) are modified, this graph does not change, which makes it easy to produce very quickly several images. Methods for computing the Fresnel factor are given in appendix.

#### Résumé

Cet article décrit une mise en œuvre, en deux phases, d'un modèle d'éclairement global dérivé de la physique. Ce modèle utilise un modèle physique de réflexion, des densités spectrales d'énergie et ne fait aucune hypothèse sur l'aspect spéculaire des matériaux utilisés. Les scènes sont discrétisées en points et non en carreaux, ce qui permet d'utiliser n'importe quel type de surface, sans avoir à les subdiviser en petits carreaux planaires. Une structure de données, appelée graphe de visibilité, est introduite afin de déterminer efficacement la visibilité entre les points échantillons de la scène. Ce graphe reste invariant lorsque les propriétés photométriques des surfaces (réflexion, transmission, rugosité, énergies émises par les sources) sont modifiées, ce qui permet de produire rapidement plusieurs images d'aspect différent. Enfin, des méthodes de calcul du facteur de Fresnel sont données en annexe.

### 1 Introduction

Photosimulation consists in producing highly realistic images. The realistic aspect of materials can be simulated only with the use of physics-based reflection and transmission models. Such models have been introduced in [11, 17, 22]. To accurately evaluate the illumination of synthesized scenes, a global model is required. The implementation of this global model can be performed according to three approaches: one-pass methods [24, 21, 25, 19], two-pass methods [28, 30], or multi-pass methods [26, 8]. The one-pass methods perform all the illumination computations independently of the view point, allowing then a fast rendering of the same scene from different view points. However, these methods need a large amount of memory to store data. Another drawback is the aliasing defects due to sharp variations of specular reflections and specular transmissions. To avoid these defects, a very dense sampling of the scene is indispensable, which would significantly increase the data to be stored.

In the two-pass methods, the diffuse and specular components (from reflection or transmission) are computed separately; the notion of form factors are then extended to account for the specular effects contributing to the global diffuse component. In our opinion, these methods seem efficient since they offer a good realism and a non prohibitive computing time.

Even though the multi-pass methods are better suited to the rendering of caustic effects, they are very time consuming since they involve several passes: Monte Carlo path tracing, light tracing, progressive refinement radiosity...

For the reasons quoted above, the global model described in this paper has been implemented according to a twopass method. It uses a physics-based reflection model as well as a transmission model. In this model, the light powers emitted, reflected or refracted are represented by their spectral distribution, the materials are characterized by their spectral reflectance (Fresnel factor) and their spectral transmittance as well as their microscopic roughness.



<sup>\*</sup>CSTB, Eclairage et Colorimétrie, 11 rue Henri-Picherit, 44300 NANTES Cedex 03, FRANCE. Tel: +33 40 37 20 00, Fax: +33 40 37 20 40

As suggested in [20], all the light powers are sampled at four wavelengths. The used reflection model is Cook's and Torrance's model [11]. Moreover, our global model does not make any assumption on the specular behavior of materials.

As pointed out hereafter, in our method, the scene is discretized into points instead of small patches, which avoids the breaking down of all surfaces into small patches. So, any kind of object can be used.

This paper addresses the following subjects. First, the global lighting model is presented as well as the reflection and transmission models. Then, we describe the different processings involved by our implementation: discretization of the scene, expression of the discretized light energy balance equation, and the two passes. In section 4, a data structure, named *visibility graph*, is described in details. It will be shown that this data structure reduces drastically the amount of time needed for computing the visibility between the sample points of the scene. Finally, some experimental results are given, and a comparison with other methods is made. Methods for computing the Fresnel factor are given in appendix.

### 2 The Global Model

### 2.1 System of Light Energy Balance Equations

A global illumination model must take into account all the reflections and refractions within the scene. To describe the mechanism of light transport we use the model introduced in [5, 6]. This model consists of a set of equations which express (in terms of *radiance* since it is the quantity the eye is sensitive to) the radiance of a point  $P_i$ in the direction of  $P_j$  when illuminated by all the surfaces  $S_k$  (see figure 1):

$$L_{\lambda}(P_i, P_j) = h(P_i, P_j) *$$
(1)

 $\left[L_{\lambda}^{E}(P_{i}) + \sum_{k} \int_{S_{k}} R_{\lambda}(P_{k}, P_{i}, P_{j}) L_{\lambda}(P_{k}, P_{i}) G(P_{k}, P_{i}) dS_{k}\right]$ where

wnere

- $L_{\lambda}(P_i, P_j)$  is the radiance of  $P_i$  as seen from  $P_j$  (emitted light power per unit surface and unit projected solid angle),
- $h(P_i, P_j)$  is the visibility function,
- $L_{\lambda}^{E}(P_{i})$  is the self-emitted radiance,
- $R_{\lambda}(P_k, P_i, P_j) = s_i R_{\lambda}^s(P_k, P_i, P_j) + d_i R_{\lambda}^d(P_i)$  is either the bidirectional reflectance [11], or the bidirectional transmittance [23],
- $L_{\lambda}(P_k, P_i)$  is the radiance of  $P_k$  as seen from  $P_i$ ,
- G(P<sub>k</sub>, P<sub>i</sub>) = cos α<sub>i</sub> cos β<sub>k</sub>/||P<sub>k</sub> P<sub>i</sub>||<sup>2</sup> is a purely geometrical term,
   λ is a given wavelength.

The system made up of these equations is called *system* of *light energy balance equations*. Solving this system will provide the global radiance at each point of the scene.



Figure 1: Geometry of light transport mechanism

### 2.2 Reflection Model

The used reflection model is the one proposed by Cook and Torrance [11]. With this model, the reflected light depends on the wavelength, the incidence angle, the roughness parameter, and the surface refractive index (this index is a complex number for metallic materials). This model takes into account the polarization of the light, the roughness and the masking/shadowing of the materials. Let us briefly review this model.

This model is expressed as:

$$R = sR_s + dR_d$$
 with  $s + d = 1$ 

where  $R_d$  and  $R_s$  are respectively the diffuse and specular components, d and s are the proportions of the incident light which give rise to the diffuse and specular components respectively.

 $R_d$  is independent of the incident angle, and can be approximated by  $\frac{F(\lambda,0)}{\pi}$  [11], where  $F(\lambda,0)$  is the Fresnel factor for a normal incidence.

 $R_s$  accounts for the roughness as well as for the masking/shadowing effects, and is expressed as:

$$R_s = \frac{1}{4\pi} \frac{F(\lambda, \theta).D.G}{\cos \theta_i \cos \theta_r},$$

where  $F(\lambda, \theta)$  is the Fresnel factor,  $\theta_i$  is the incidence angle (direction  $D_i$ ),  $\theta_r$  the reflection angle (direction  $D_r$ ) and  $\theta$  equals half of the angle  $(D_i, D_r)$ . G is the masking/shadowing function, and D models the roughness effect. In our implementation, D is the Beckman function. The Fresnel factor is given by the Fresnel formula. In appendix, we show how this factor can be computed efficiently (even in case of metallic materials).

### 2.3 Transmission Model

So far, no physics-based transmission models have been proposed in the literature, but only an empirical one [14]. Rather than using an empirical transmission model, it is more realistic, for each material, to use transmittance values experimentally obtained with the help of a spectrophotometer [12].





Figure 2: Sampling a polygon into points

However, if any transmission model exists, it can be easily integrated in our implementation. For this reason, at the present time, only ideal specular refraction is available in our method. In case of ideal specular refraction,  $R_s$  is no more than  $1 - F(\lambda, \theta)$ , and s = 1.

### 3 The Method

### 3.1 Discretizing the Scene Into Points

#### 3.1.1 Motivation

Any method used to implement a global illumination model requires a discretization of the scene either into patches or points. In our method, we have chosen to discretize the scene into points for the following reasons:

- we have to evaluate the radiance for each point of the scene in one direction (radiance is directional),
- to evaluate the radiance of a point, it is not necessary to consider all the points of the scene. Indeed, since the radiance is an integral, we can evaluate it by taking a certain number of samples of the variables of the integrand (Gauss or Monte Carlo methods),
- visibility between two points can be easily computed by ray tracing,
- extended form factors, in presence of non ideal specular surfaces, can be easily evaluated by tracing rays from point to point, while their evaluation is very difficult when using patches,
- several kinds of surface can be sampled into points: polygons, spheres, cylinders, cones, parametric surfaces etc.. Subdivision into patches can then be avoided, which allows the use of different kinds of geometric models.

#### 3.1.2 The Discretization Method

In our present implementation, the scene is made up of a collection of convex quadrilaterals. The discretization process consists in recursively subdividing each quadrilateral in four subsurfaces. The recursion stops when the area of a subsurface is below a certain threshold fixed by the user. Once this subdivision has been accomplished, a sample point is placed at the center of each subsurface. With each sample point is associated the area  $\delta S$  of the surface containing it. These areas  $\delta S$  are used to evaluate the solid angles between two sample points. This process is illustrated by figure 2. We preferred to consider the centers of the subsurfaces as sample points, rather than the vertices, so as to avoid undesirable effects along the edges shared by two surfaces. Note that a better subdivision would be to add a second threshold for the differential solid angles between two samples as suggested in [15, 16].

### 3.2 The Discretized Light Energy Balance Equation

Since the scene is discretized into points, only the pointto-point light contributions have to be evaluated. In order to discretize equation (1), we exploit the fact that with each sample point of a surface  $S_k$  is associated a surface area  $\delta S_k$ . Equation (1) becomes then:

$$L_{ij} \approx h_{ij} \left[ L_i^E + \sum_{k=1}^N R_{kij} G_{ki} L_{ki} \delta S_k \right]$$
(2)

where N is the number of sample points in the environment. Using this point sampling and separating the diffuse and specular reflections, we obtain:

$$L_{ij} = h_{ij} \left[ L_i^E + L_i^d + L_{ij}^s \right]$$
(3)

$$L_i^d = \sum_{k=1}^N d_i R_i^d G_{ki} L_{ki} \delta S_k = \sum_{k=1}^N \mathcal{D}(L_{ki}) \qquad (4)$$

$$L_{ij}^{s} = \sum_{k=1}^{n} s_{i} R_{kij}^{s} G_{ki} L_{ki} \delta S_{k} = \sum_{k=1}^{n} \mathcal{S}(L_{ki}), \quad (5)$$

where  $L_i^D$  is the diffuse component of the reflected light,  $L_i^S$  the specular component and n is the number of sample points included in the specular reflection cone (or specular transmission cone) which bounds the specular component  $R^s$  of R. The angle of such cones depends of the roughness of the materials. The more important the roughness, the larger the angle. Such cones are defined by an axis which is the perfect reflection (or transmission) direction, and by an angle which depends on the physical properties of the materials (see figure 3). In our case, the reflection cone bounds the Beckman function D, since this latter models the roughness. In case of perfectly specular materials, the reflection cone is reduced to the perfect reflection direction. As no sample point will lie along this direction, we select the sample point which is closest to this direction. Note that the operators  $\mathcal{D}$  and  $\mathcal{S}$  are the global diffuse and global specular operators, respectively.

Since transmission is treated exactly analogously to reflection, we will omit it for now.

The diffuse component is evaluated by adding the contribution of all points in the scene to a given point, while the specular component takes into account only the points whose contribution will be significant, i.e. the points included in the reflection cone.





Figure 3: the reflection cone

#### 3.3 First Pass

#### 3.3.1 Extended Form Factors

Since a reflected light can be separated into diffuse and specular components [5], the two-pass method consists of two main steps: the first one is a view independent computation of the *Global Diffuse Radiance* while the second evaluates the view dependent *Global Specular Component*. Let us rewrite equation (1):

$$L_{ij} = h_{ij} \left[ L_i^E + \sum_{k=1}^N (d_i R_i^d + s_i R_{kij}^s) G_{ki} L_{ki} \delta S_k \right]$$
$$= h_{ij} \left[ L_i^E + \sum_{k=1}^N d_i R_i^d G_{ki} L_{ki} \delta S_k + \sum_{k=1}^n s_i R_{kij}^s G_{ki} L_{ki} \delta S_k \right]$$
$$= h_{ij} \left[ L_i^E + \sum_{k=1}^N \mathcal{D}(L_{ki}) + \sum_{k=1}^n \mathcal{S}(L_{ki}) \right]$$

The global diffuse radiance  $L_i^d$  of a point is then given by: {

$$L_{i}^{d} = L_{i}^{E} + \sum_{k=1}^{N} d_{i} R_{i}^{d} G_{ki} L_{ki} \delta S_{k} = L_{i}^{E} + \sum_{k=1}^{N} \mathcal{D}(L_{ki}).$$
(6)

The expression of  $L_i^d$ , as function of  $L_j^d$  is then:

$$L_i^d = L_i^E + \sum_{j=1}^N (\mathcal{D} + \mathcal{D}(\sum_{k=1}^n \mathcal{S}) + \mathcal{D}(\sum_{l=1}^n \mathcal{S}(\sum_{m=1}^n \mathcal{S})) + \dots) L_j^d.$$

The term  $\left[\mathcal{D} + \mathcal{D}(\sum_k S) + \mathcal{D}(\sum_l S(\sum_m S)) + ...\right]$  is called *Extended Form Factor* and is named  $EFF_{ji}$  by analogy with radiosity [28, 6].

It is the proportion of diffuse light emitted by the surface  $S_j$  that contributes to the global diffuse radiance of the point  $P_i$  (see figure 4). Using these terms we obtain a system of N equations of N unknowns:

$$L_i^d = L_i^E + \sum_j \left[ EFF_{ji} * L_j^d \right] \tag{7}$$



Figure 4: Extended form factor  $FFE_{ii}$ 

This system can be solved (for each wavelength) by using the same methods as for radiosity (complete or iterative solution [9]).

#### 3.3.2 Algorithm

The following algorithm computes the contribution of one given point to all its environment. This algorithm uses a shooting process since it allows a progressive solution. /\* statements \*/

/\* matrix of spectral extended form factors \*/ spectrum  $EFF_{\lambda}[N][N]$ 

/\* N is the number of sample points \*/

/\* spectrum initialized to 1.0 \*/
spectrum spec1 = {1.0, ..., 1.0}

/\* computes the j<sup>th</sup> column of the extended form
factors matrix \*/
EvaluateEFF(j)

for all points 
$$P_i$$
 {  
 /\* geometrical term \*/  
 compute  $FF_{ji} = h_{ji}G_{ji}\delta S_j$   
 /\* for each spectrum sample \*/  
 $EFF_{\lambda}[j][i] = EFF_{\lambda}[j][i] + FF_{ji} * d_i R_{\lambda i}^d$   
 if  $(s_i \neq 0)$  { /\* specular surface \*/  
 GlobSpecOp(j, j, i, spec1, 1)  
 }  
}

The procedure EvaluateEFF evaluates the direct contribution of the global diffuse radiance of  $P_j$  to the global diffuse radiance of all the points of the scene. All points being illuminated by  $P_j$  and belonging to a non perfectly diffuse surface emit the specularly reflected light to the environment. This is made possible thanks to the following procedure GlobSpecOp.



}

```
/* Global Specular Operator*/
GlobSpecOp(j, 1, i, ΔEFF<sub>λ</sub>, 1g)
/* j: emitting point */
/* i: last point met */
/* l: previous point in light path */
/* ΔEFF<sub>λ</sub>: cumulated contribution */
/* of P<sub>j</sub> to P<sub>i</sub> via P<sub>l</sub> */
/* lg: length of path from P<sub>j</sub> to P<sub>i</sub> */
{
/* light is emitted by P<sub>l</sub> */
/* and reflected by P<sub>i</sub> towards points P<sub>k</sub>*/
/* geometrical term */
```

```
compute FF_{li} = G_{li}\delta S_l
for all points P_k \in \text{reflection cone} \{
    /* geometrical term */
    compute FF_{ik} = h_{ik}G_{ik}\delta S_i
    /* for each wavelength */
    \Delta EFF_{\lambda} = \Delta EFF_{\lambda} * FF_{li} * s_i R^S_{\lambda lik}
    EFF_{\lambda}[j][k] = EFF_{\lambda}[j][k]
        + \Delta EFF_{\lambda} * FF_{ik} * d_k R^d_{\lambda k}
    if ((lg < lgmax)
        and(\Delta EFF_{\lambda} > EFFMIN_{\lambda})
        and(s_k \neq 0)) {
        GlobSpecOp(j, i, k, \Delta EFF_{\lambda}, lg+1)
    }
}
```

#### 3.4 Second Pass

}

In the second pass (which is view dependent) the global specular operator is evaluated by means of a distributed ray tracing [10]. Note that this step does not entail the shooting of rays towards light sources (shadow rays), since the global diffuse component of the sample points (of the scene) is already available. Moreover, the intersection between a shot ray and the scene results in a point which may not be a sample point. However, the global diffuse component of the radiance at this intersection point can be interpolated by using the diffuse spectral radiances of the four closest sample points computed at the first pass as done in [2, 18].

### 4 Visibility Calculation

This section shows how the visibility between two sample points is performed. To evaluate the visibility between a pair of sample points  $P_i$  and  $P_j$ , a ray is cast from  $P_i$ toward  $P_j$ . If this ray hits one object before reaching  $P_j$ , then the two sample points do not see each other. This process entails a large amount of ray-object intersections and needs to be accelerated. To this end, two data structures are used: spatial subdivision and visibility graph.



Voxel2 and Voxel4 are fully hidden Voxel1 and Voxel4 are partially hidden: the subgraph (p11,p31),(p11,p32), (p12,p31),(p12,p32) is stored

Figure 5: Voxel-to-voxel visibility information

### 4.1 Spatial Subdivision

The parallelepipedic bounding volume of the scene is recursively subdivided by planes aligned with the coordinate system axes. Each slicing plane subdivides a space into two equal sized subspaces. This subdivision results in a set of unequal sized subspaces which, from now, are called voxels. With each voxel is associated a list of sample points located in this voxel. This recursive subdivision stops either when the number of objects (which are convex quadrilaterals in our current implementation) intersecting the current voxel is below a certain threshold, or when the maximum level of subdivision is reached.

To apply Amantides's traversal algorithm [1], a spatial index SI is used. This spatial index is a 3D grid, whose each element SI[i, j, k] is a pointer to a voxel (see [4] for more details).

In contrast to a uniform grid, our spatial subdivision into unequal sized voxels allow to reduce the amount of memory needed to store the visibility graph described hereafter.

Note that this spatial subdivision is also used in the second pass to calculate the global specular component by ray tracing.

### 4.2 The Visibility Graph

#### 4.2.1 The Graph

Since the scene is sampled into points, it seems worthwhile to build a visibility graph giving a boolean visibility information for all pairs of points, instead of building a valuated graph whose complexity is  $O(N^3)$  for each wavelength as done by Buckalew [7].

Graphics Interface '92

323



Figure 6: Visibility between two voxels

This visibility graph is very attractive but requires a large memory. To cope with this problem of memory size, we propose the following visibility strategy. Since the scene is subdivided into voxels, we build a visibility graph giving a visibility information between each pair of voxels instead of sample points. The visibility test checks how each voxel is visible from the other voxels. This test fires rays between the two voxels. Each ray corresponds to a pair of sample points belonging to different voxels as shown in figure 6.

The data structure used to store the visibility graph is :

```
Nedges = N(N+1)/2;
/*where N is the number of voxels*/
typedef struct /*edge data structure*/
{
boolean visibility;
object *buf-int;
booleanmatrix *voxel-graph;
}Edge;
```

/\*visibility graph data structure\*/
Edge Visibility-Graph[1..Nedges];

The nodes of the visibility graph are voxels, while its edges are elements of the linear array *Visibility-Graph[]*. The field voxel-graph is a pointer to a boolean matrix storing the visibility information between the points of the two voxels of an edge.

Let i and j be the two voxels of edge k. Three cases can be considered:

- 1. all points of voxel i are visible from all points of j. Visibility-Graph[k].visibility = true; Visibility-Graph[k].buf-int = null; Visibility-Graph[k].voxel-graph = null;
- 2. the sample points of i do not see those of j. Visibility-Graph[k].visibility = false; Visibility-Graph[k].buf-int = null; Visibility-Graph[k].voxel-graph = null;
- 3. the sample points of i see only a part of those of

j (figure 5). In such a case we decide to store the visibility information between all pairs of points included in the two relevant voxels as well as the pointer to one object lying between these two voxels.

Visibility-Graph[k].visibility = false; Visibility-Graph[k].buf-int = pointer-object; Visibility-Graph[k].voxel-graph = pointer-matrix;

Note that the field buf-int of the data structure Edge plays an important role. Indeed, during the visibility test between two voxels, as soon as a fired ray intersects objects lying between the two voxels, only the pointer to the closest object is stored in the field buf-int of Edge. Due to the spatial coherence, the next fired ray has a great probability to intersect the same closest object between the two voxels. Consequently, this next fired ray will be checked for intersection with only this closest object, the pointer of which is already in buf-int, which saves a significant amount of computation. This approach seems to be a simplified version of the light buffer method [13].

In most scenes we have treated in our experiments, this strategy appeared rather efficient but we must keep in mind that the real memory complexity of this graph is always  $O(N^2)$ .

# 4.2.2 Using the Visibility to Improve the Scene Discretization

During the evaluation of the visibility graph, the distance between each pair of points is computed. If the distance between two points is small compared to their associated surface area  $\delta S$  (important solid angle), the corresponding surface elements are locally subdivided. We obtain then new sample points with smaller associated surface areas. Thereby, the solid angles between these two points become smaller, which makes the computation more accurate.

#### 4.2.3 Other Advantages

As said above, the visibility graph contains only purely geometric information, independent of the photometric properties of the objects. This allows to modify these properties (reflectance, transmittance, roughness, selfemittance...) while keeping the same visibility graph. Indeed, to obtain new values of radiance, only a graph traversal is needed. Moreover, when a few objects are moved, only a small part of the visibility graph has to be modified. This graph might be updated with a rapid incremental method. This method is currently under investigation.

### 5 Results

We express the cost of the evaluation of the extended form factors matrix in term of numbers of calls to the visibility



number of voxels	102
number of empty voxels	5
number of pairs of points	16076090
number of pairs of visible voxels	100
number of pairs of hidden voxels	1009
number of pairs	4042
of partially hidden voxels	
number of pairs of points stored	5 889 008
graph computation time	39mn 49sec.

Table 1: Cost of the visibility graph

	With Graph	Without Graph
Time (seconds)	48mn 33sec	3h 24mn 12sec
Number of visibility	8038045	42 433 940
computations		

 Table 2: Computation of the matrix of extended form

 factors

function  $h_{ij}$ . The cost of our method is given when the visibility graph is used, and when it is not. The results obtained are in favor of the use of this graph.

Our test scene is made up of 134 polygons. The materials of the objects of the scene are gold, sand, concrete, brown stone, blue and green enamel. All these materials are perfectly diffuse except gold (leg of the table) and silver (mirror) whose parameters are: s = 0.9, d = 0.1 and the roughness coefficient m = 0.3 (Beckmann term) for gold, and s = 0.9, d = 0.1 and m = 0.05 for silver. The scene includes two primary light sources which are normalized D6500 white sources. The sampling of the scene results in 4010 points, 196 of which lie on specular surfaces. To emphasize the influence of specular materials on the global diffuse radiance, our test scene was processed according to three different ways, giving the three following images:

- image 1 (figure 7): all the materials are assumed to be perfectly diffuse,
- image 2 (figure 8): scene containing diffuse and specular materials, image resulting from the first pass,
- image 3 (figure 9): scene containing diffuse and specular materials, final image obtained after the two passes.

Note that the light spots near the leg of the table are the global diffuse component due to the specular properties of this golden leg. The spot light near the door is due to specular reflection on the silver mirror.

The amount of memory required to store (table 1) the visibility graph has been drastically reduced thanks to our non uniform spatial subdivision, whereas this amount is very important for a spatial subdivision into a regular 3D grid.

Among the 8.038.045 visibility calculations to be performed, only 1.394.017 of them are actually made. Indeed, the visibility computations corresponding to the following cases are avoided:

- sample points of the same surface;
- the angle formed by the ray direction and the normal at a sample point is greater than 90 deg;
- the buffer *buf-int* is used to evaluate the visibility function.

## 6 Comparisons with Other Algorithms

In contrast to our model, the global illumination algorithms described in [29, 28, 18] use an empirical reflection model, a trichromatic approximation, and an ideal specular reflection. Even though the algorithm in [27] uses a new physics-based reflection model, it is limited to ideal reflection. As for the multi-pass ones [26, 8], they seem more suited for rendering caustics but are very time expensive compared to the two-pass methods.

### 7 Conclusion

Unlike most of the models already implemented, our illumination model accounts for a physics-based reflection model, spectra instead of a trichromatic approximation, the spectral reflectance and transmittance of materials as well as color science. In our implementation, the scene is sampled into points instead of small patches. To prove that point sampling is correct, we have generated one image with that kind of sampling. It has been compared with the image of the same scene generated by a technique based on a discretization into patches. The visual results seem very similar. This point discretization offers the advantage of evaluating, very easily, the extended form factors when no assumption is made on the specular behavior of materials.

Even though the visibility graph requires an important memory size, it significantly reduces the synthesis time, and in addition, it is well suited to an adaptive point discretization that improves the precision of the solid angle calculations, which avoids thus all artifacts.

### References

- J. Amanatides and A. Woo. A fast voxel traversal algorithm for ray tracing. In EUROGRAPHICS'87 Conference Proceedings, pages 3-10, August 1987.
- [2] J. Arvo. Backward ray-tracing. In Course of the SIGGRAPH'86. Developments in Ray-Tracing, 1986.
- [3] M. Born and E. Wolf. Principles of optics. Pergamon press, 1970.
- [4] K. Bouatouch, M.O Madani, T. Priol, and B. Arnaldi. A new algorithm of space tracing using a CSG model. In EUROGRAPHICS'87 Conference Proceedings, pages 65-78, August 1987.



- [5] C. Bouville and K. Bouatouch. A unified approach to global illumination models. In *PIXIM'89 Confer*ence, pages 250-263, September 1989.
- [6] C. Bouville, K. Bouatouch, P. Tellier, and X. Pueyo. Theoretical analysis of global illumination models. In *Photorealism in Computer Graphics*, pages 57-71, EurographicSeminars, Springer-Verlag, 1992.
- [7] C. Buckalew and D. Fussel. Illumination networks: fast realistic rendering with general reflectance functions. Computer Graphics, 23(3):89-98, July 1989.
- [8] S. E. Chen, H. E. Rushmeier, G. Miller, and D. Turner. A progressive multi-pass method for global illumination. *Computer Graphics*, 25(4):165-174, August 1991.
- [9] Michael F. Cohen, Shenchang E. Chen, John R. Wallace, and Donald P. Greenberg. A progessive refinement approach to fast radiosity image generation. *Computer Graphics*, 22(4):75-84, August 1988.
- [10] R. L. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. *Computer Graphics*, 18(3):165-174, July 1984.
- [11] R.L. Cook and K.E. Torrance. A reflectance model for computer graphics. ACM transactions on graphics, 1(1):7-24, January 1982.
- [12] M. Garcia, M. Perraudeau, and P. Chauvel. Experimental measurements of reflectance and transmittance with a spectrophotometer. February 1992. CSTB, private communication.
- [13] E.A. Haines and D.P. Greenberg. The light buffer: a shadow testing accelerator. *IEEE Computer Graphics and Applications*, 6(9):6-16, September 1986.
- [14] A. Roy Hall and Donald P. Greenberg. A testbed for realistic image synthesis. *IEEE Computer Graphics* and Applications, 3(8):10-20, November 1983.
- [15] P. Hanrahan and D. Salzman. A rapid hierarchical radiosity algorithm for unoccluded environments. In *Photorealism in Computer Graphics*, pages 151–169, EurographicSeminars, Springer-Verlag, 1992.
- [16] P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. Computer Graphics, 25(4):197-206, August 1991.
- [17] X. D. He, K. E. Torrance, F. X. Sillon, and D. P. Greenberg. A comprehensive physical model for light reflection. *Computer Graphics*, 25(4):175-186, August 1991.
- [18] P.S. Heckbert. Adaptative radiosity textures for bidirectional ray tracing. Computer Graphics, 24(4):145-154, August 1990.
- [19] D. S. Immel, M. F. Cohen, and D. P. Greenberg. A radiosity method for non-diffused environments. *Computer Graphics*, 20(4):133-142, July 1986.
- [20] Gary W. Meyer. Wavelength selection for synthetic image generation. Computer Vision, Graphics, and Image Processing, 41:57-79, 1988.

- [21] S.P. Mudur and S.N. Pattanaik. Multidimensional illumination functions for visualization of complex 3d environment. Visualization and Computer Animation, 1(2):49-58, 1990.
- [22] P. Poulin and A. Fournier. A model for anisotropic reflection. Computer Graphics, 24(4):273-282, August 1990.
- [23] H. Rushmeier and K. Torrance. Extending the radiosity method to include reflecting and translucent materials. ACM Transaction on Graphics, 9(1):1-27, January 1990.
- [24] B. Le Saec and C. Schlick. A progressive ray-tracingbased radiosity with general reflectance functions. In *PhotoRealism in Computer Graphics*, pages 101–113, EurographicSeminars, Springer-Verlag, 1992.
- [25] M. Shao, Q. Peng, and Y. Liang. A new radiosity approach by procedural refinements for realistic image synthesis. *Computer Graphics*, 22(4):93-101, August 1988.
- [26] P. Shirley. A ray tracing method for illumination calculation in diffuse-specular scenes. In *Graphis In*terface Conference Proceedings, pages 205-212, May 1990.
- [27] F. 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, August 1991.
- [28] F. Sillion and C. Puech. A general two-pass method integrating specular and diffuse reflection. *Computer Graphics*, 23(3):335-344, July 1989.
- [29] J. R. Wallace, M. F. Cohen, and D. P. Greenberg. A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity methods. *Computer Graphics*, 21(4):311-320, August 1987.
- [30] J. R. Wallace, K. A. Elmquist, and E. A. Haines. A ray tracing algorithm for progressive radiosity. *Computer Graphics*, 23(3):315-324, July 1989.
- [31] R.C. Weast, D.R Lide, M.J. Astle, and W.H. Beyer. CRC Handbook of Chemistry and Physics. CRC Press, Inc. Boca Raton, Florida, 1989-1990.
- [32] G. Wyszecky and W. S. Stiles. Color Science, Concepts and Methods, Quantitative Datas and Formulas. J. Willey and sons, 1982. 2nd Edition.

### APPENDIX

### Computing the Fresnel factor

We have implemented two methods to efficiently evaluate  $F(\lambda, \theta)$ .



Graphics Interface '92

#### First method

In several books [32, 3, 31], we can find, for several materials, Fresnel factor curves  $F(\lambda, 0)$  for normal incidence, as well as the refraction index  $\hat{n}$  for the wavelength  $\tilde{\lambda} = 589$  (Sodium D lines) which corresponds to the center of the visible bandwith. Given these data,  $F(\lambda, \theta)$  can be approximated [11], for each wavelength, by:

$$F(\lambda_i, \theta) = F(\lambda_i, 0)$$

$$+\left(F(\lambda_i,\frac{\pi}{2})-F(\lambda_i,0)\right)\frac{F(\tilde{\lambda},\theta)-F(\tilde{\lambda},0)}{F(\tilde{\lambda},\frac{\pi}{2})-F(\tilde{\lambda},0)},$$

where  $F(\tilde{\lambda}, \theta)$  is given by the Fresnel formula for  $\hat{n}$ .

#### Second method

In [31], for several materials, values of the refraction index are given for a certain number of wavelengths. In this case,  $F(\lambda, \theta)$  can be exactly expressed with the Fresnel formula.

### Storage of $F(\lambda, \theta)$

Knowing the expression of  $F(\lambda, \theta)$ , we can precompute it for each sample wavelength and for different values of  $\theta$ (20 seem enough). These values allow to create a lookup table, from which any  $F(\lambda, \theta)$  can be computed by a simple linear interpolation.







Figure 7: Image 1: result of the first pass; perfectly diffuse materials



Figure 8: Image 2: result of the first pass; diffuse and specular materials



Figure 9: Image 3: result of the two passes



Graphics Interface '92