# A General Two-Pass Method Integrating Specular and Diffuse Reflection.

*François Sillion, Claude Puech*

Laboratoire d'Informatique de l'Ecole Normale Supérieure
U.R.A. 1327, CNRS

## Abstract

We analyse some recent approaches to the global illumination problem by introducing the corresponding *reflection operators*, and we demonstrate the advantages of a two-pass method. A generalization of the system introduced by Wallace *et al.* at Siggraph '87 to integrate diffuse as well as specular effects is presented. It is based on the calculation of *extended form-factors*, which allows arbitrary geometries to be used in the scene description, as well as refraction effects. We also present a new sampling method for the calculation of form-factors, which is an alternative to the *hemi-cube* technique introduced by Cohen and Greenberg for radiosity calculations. This method is particularly well suited to the extended form-factors calculation. The problem of interactive display of the picture being created is also addressed by using hardware-assisted projections and image composition to recreate a complete specular view of the scene.

CR Categories and Subject Descriptors: **I.3.3 [Computer Graphics]**: Picture/Image Generation - Display Algorithms. **I.3.7 [Computer Graphics]**: Three-Dimensional Graphics and Realism.

Additional Key Words and Phrases: radiosity, interreflection, two-pass method, extended form factors, z-buffer, progressive refinement, global illumination, ray tracing.

## 1 Introduction

The problem of light interreflection has been one of the main issues for realistic image synthesis during the last few years. It is now widely known that local lighting models are not sufficient to compute an accurate distribution of light within an environment [13] [1] [5]. The multiple reflections of light on the objects in the scene account for a large part of the total distribution of light, and a global solution must therefore be computed, for the intensity of light at some point may depend on the intensity at any other point. The first global models, ray tracing and radiosity, made strong assumptions about the reflection process, namely that it is either purely specular or purely diffuse.

During the last three years, some advanced models have been introduced that allow arbitrary reflection modes to be used. We review these models briefly in section 2, and show how to describe them using a common formulation, similar to the one introduced by Kajiya with the *rendering equation* [11]. This leads to a new computational system (section 3) extending the work of Wallace *et al.* [16]. It is a general two-pass system that permits the inclusion of refraction among the effects modeled, and removes the previous restriction that all specular surfaces must be planar mirrors. We then present a sampling method using adaptive subdivision (section 4) particularly suited to our two-pass method, and show that it is an interesting alternative to the classical hemi-cube technique [3] in the diffuse radiosity case as well. Finally, we show in section 5 how to produce pictures integrating a complete specular behavior at interactive rates, using multiple hardware z-buffers.

LIENS : 45, rue d'Ulm. 75230 Paris Cedex 05. FRANCE.
Tel : (33) (1) 43 29 12 25 ext. 32-16. Fax : (33) (1) 46 34 05 31.
e-mail : sillion@frulm63.bitnet, puech@frulm63.bitnet.

 ACM-0-89791-312-4/89/007/0335 $00.75

## 2 A reformulation of previous models using the *rendering equation*

At the Siggraph '86 conference, Kajiya introduced an equation describing the transfer of light between surfaces in an environment [11]. We shall here reformulate some recent models within this framework, and introduce different kinds of reflection operators, corresponding to the assumptions made by these models.

### 2.1 The equation

Kajiya's rendering equation is the following :

$$I(x,x') = g(x,x')[\varepsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx''].$$

(We use here the exact formulation of Kajiya's paper, and we shall not discuss this formulation. The reader is invited to refer to [11] for further details and a discussion of its validity). Let us just recall what the different terms of this equation mean :

The domain $S$, over which the integral is calculated, is the union of the surfaces of all objects composing the scene. $I(x,x')$ is the transport intensity from point $x'$ to point $x$, $g(x,x')$ is a visibility function between $x$ and $x'$, which value is 0 if $x$ and $x'$cannot see

each other, and $\frac{1}{d(x,x')^2}$ otherwise. $\varepsilon(x,x')$ is the transport emittance from $x'$ in the direction of $x$. $\rho(x,x',x'')$ is a bi-directional reflectance function at point $x'$, with respect to the directions of $x$ and $x''$.

### 2.2 The global reflection operator

As Kajiya states it in his paper, one can define a reflection operator $\mathcal{R}$ as an integral operator which describes the effects of the reflection on all surfaces on a given light distribution, and express the rendering equation as :

$$I = g\,\varepsilon + \mathcal{R}I$$

(please note that the visibility term is integrated in the reflection operator. This means that the reflected light appears only at points that can see the reflector). The rendering equation can then be formally inverted to give an expression which makes apparent the contributions of the successively scattered terms.

$$I = \sum_{n=0}^{\infty} \mathcal{R}^n g\,\varepsilon$$

### 2.3 Direct solution using Monte-Carlo integration

Stochastic sampling, as introduced in the computer graphics field by Cook *et al.* [4], gives a way to actually evaluate the reflection integral, which was further investigated by Kajiya, in the same paper where he coined the rendering equation. This is an elegant solution because it solves the entire equation, for all directions converging to the viewpoint, but it involves sampling a huge number of directions, if complex reflective behaviors are to be modeled. Furthermore, the solution is dependent on the light sources in the scene. The problem is to find a general law for the choice of the samples, solving the tradeoff between accurate sampling and computation time. The method introduced by Ward *et al.* [17] at Siggraph '88 can be used to reduce the number of samples at each stage, as it concentrates on specular, rapidly-varying effects, calculating the slowly-varying "ambient" effects less often.

### 2.4 Radiosity-based solution

At the same '86 conference, Immel *et al.* [10] presented another general solution of a similar equation, based on the previous radiosity method. The basic idea of radiosity is to discretize the space of variables in the transfer equations, thus transforming the integral equation into a system of linear equations. This involves computing the matrix elements of the reflection operator. The solution of the linear system is inherently a global solution, and the good points of radiosity are that the geometric dependency of the matrix elements needs not to be recomputed if only lighting conditions are changed, and that the solution is independent from the viewpoint, as it gives an intensity value for each discretized sample.

In the classical radiosity method [8],[3], the discretization is performed by defining an intensity value (radiosity) for each of a number of surface patches. It is then assumed that the directional distribution of the emitted light is lambertian (diffuse).

The method of Immel *et al.* is more general, because it removes the restriction that surfaces must be lambertian reflectors. This is done by taking as a discrete unit a couple (patch, direction), with a finite number of patches, and a finite number of directions. The matrix coefficients of operator $\mathcal{R}$ are calculated during the solution process, using the visibility information provided by the usual hemi-cube [3, see also section 4.1 for a definition of the hemi-cube].

Immel's solution is thus a complete solution, like Kajiya's path tracing, but it involves computing and solving a gigantic linear system of equations. Even if the matrix is very sparse, the CPU power needed makes it unpractical for application purposes. Therefore, some hybrid solutions have been proposed, which attempt to capitalize on ray tracing or radiosity strengths.

### 2.5 Shao's progressively refined form-factors

At Siggraph '88, Shao *et al.* [15] presented a method allowing rendering of specular effects, while maintaining a relatively low CPU cost. Their method is a simplification of Immel's one where, instead of keeping track of the energy emitted by each patch for each direction, one only considers the energy emitted from a patch to another patch (thus grouping together all corresponding directions). Recall that for the diffuse radiosity this reduces further to one energy value per patch, since the distribution of this energy among the other patches is entirely specified by Lambert's law (plus the visibility information from the hemi-cubes). Shao's idea is to use the geometrical information provided by the hemi-cube as energy transfer information. The percentage of energy leaving a given patch, say $i$, for another patch $j$ is estimated by considering the geometrical relationships between patch $i$ and all other patches in the scene, which allows a directional analysis of the impinging light on $i$. In other words, one determines where the incoming light comes from, in order to decide whether it is reflected toward patch $j$ or not. Shao's use of the terminology "form factor" is somewhat misleading, although the definition matches the one for the usual form factors, since it depends on the current distribution of light in the scene, and not only on geometrical aspects. Another way to express this idea is to introduce the reflection operator modeled by such form factors. It is easy to see that each step of Shao's iterative procedure computes a "current" light distribution $I_k$, such that

$$I_k = g\,\varepsilon + \mathcal{R}_k I_k \qquad \Leftrightarrow \qquad I_k = \sum_{n=0}^{\infty} \mathcal{R}_k^n g\,\varepsilon$$

by performing a radiosity solution with a current reflection operator $\mathcal{R}_k$. The initial operator $\mathcal{R}_0$ is a diffuse (Lambertian) operator, which corresponds to the usual diffuse form factors. The form factors are re-computed at each stage, producing a new reflection operator based on the current light distribution, and the whole process is started again. The crucial point in the method is the derivation of $\mathcal{R}_k$ (the improved form factors at step $k$), from a given light distribution $I_{k-1}$.

The convergence of the process towards the correct distribution of light is established using a subtle argument in Shao's paper, and can be intuitively guessed (but not proved) since each step basically adds to the description of the light leaving a given patch the light reflected from the directions where "new" light has arrived. Thus specular propagation of light is simulated, and the contribution of each step is decreasing as the process runs.

One important thing to note is that, even if $I_k$ successfully converges towards the correct distribution, this is not the case for $\mathcal{R}_k$. The final operator $\mathcal{R}_\infty$ is different from the global reflection operator $\mathcal{R}$, as it models only light transfers that actually occur under the given lighting conditions. In other words, the effets of $\mathcal{R}_\infty$ and of $\mathcal{R}$ on the limit distribution $I_\infty$ are the same. This implies that the form factors must be re-computed when lighting conditions change, in contrast to conventional radiosity.

Although this method, like Immel's one, produces directional information about the light leaving each patch, the authors had to add a final ray tracing pass in order to accurately render the rapid changes in the specular effects across the surfaces, as seen from the eye. However, this method is not inherently a two-pass method like the ones studied later in this paper (see discussion in section 3.3).

### 2.6 Two-pass methods

First introduced by Wallace *et al.* [16] in a specific case, the two-pass approach is based on a distinction of reflection modes. The essence of the approach is to have a radiosity program calculate the diffuse part of light, and a ray tracing program calculate the specular part. Unfortunately, one cannot completely separate the computation of the diffuse and the specular components for the light, because the light itself is not diffuse nor specular; these qualifications apply in fact only to the reflection modes of the light. In other words, some quantity of light can be specularly reflected by a surface $\mathcal{S}_1$, then diffusely reflected by a surface $\mathcal{S}_2$, and so on...(figure 1).

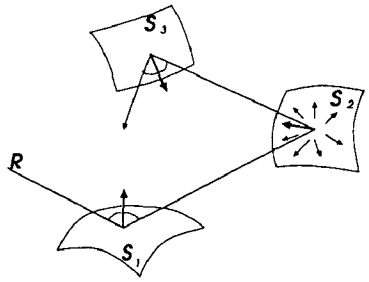



Figure 1: Light traveling along the path $R$ becomes successively "specular" (from $S_1$ to $S_2$), "diffuse" (from $S_2$ to $S_3$), and "specular" again (after $S_3$).

## 3 The extended two-pass method

We present a two-pass method, referred to as the *extended two-pass method* in the sequel, which allows all types of reflection modes to be simulated, removes the restriction in [16] that specular surfaces must be planar mirrors, and includes refraction among the set of lighting effects modeled.

### 3.1 The basic equations

In our method, we separate light reflection into two modes :

- A diffuse reflection : some part of the incident light is re-emitted according to Lambert's law.
- A specular (directional) reflection (and refraction) : some other part is re-emitted around the directions associated with the incident direction by Snell's laws.

In other words, we express the reflectance function as a sum :

$$\rho(x, x', x'') = \rho^d(x') + \rho^s(x, x', x'')$$

$\rho^d$ is the diffuse reflection coefficient at point $x'$, and $\rho^s$ is the specular (anisotropic) reflection function, which depends on the positions of points $x$ and $x''$ relative to point $x'$. For fixed points $x$ and $x'$, this specular function, as a function of $x''$, exhibits a peak around the mirrored image of point $x$ by the surface at point $x'$ (and another peak around the refracted direction). The exact form of this function needs not to be specified at this point.

Furthermore, we shall assume that all self-emission of light in the scene is purely diffuse (i.e. $\varepsilon(x, x') \equiv \varepsilon(x')$).

Under these assumptions, we can, by replacing $\rho$ by its full expression, rewrite Kajiya's rendering equation as :

$$I(x, x') = g(x, x')\beta(x') + \mathcal{T}I(x, x')$$

where $\beta$ depends only on $x'$ :

$$\beta(x') = \varepsilon(x') + \rho^d(x') \int_S I(x', x'')dx''$$

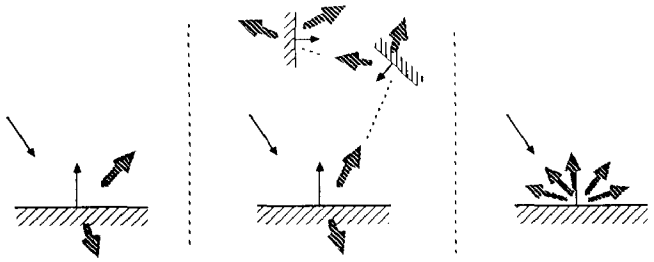
and $\mathcal{T}$, the *specular reflection-refraction operator*, is such that :

$$\mathcal{T}I(x, x') = g(x, x') \int_S \rho^s(x, x', x'')I(x', x'')dx''$$

$\mathcal{T}$ is a linear operator, transforming the light distribution $I$ into the distribution obtained by allowing one specular reflection and refraction on all surfaces in the scene (figure 2). The new equation states the relationship between the directional ($I$) and isotropic ($\beta$) distributions, and it can be formally inverted to yield :

$$I = [1 - \mathcal{T}]^{-1} g\beta \quad = \quad [\sum_{k=0}^{\infty} \mathcal{T}^k] g\beta \quad = \quad \mathcal{S} \cdot g\,\beta \qquad (1)$$

where $\mathcal{S} = \sum_{k=0}^{\infty} \mathcal{T}^k$, the *global specular operator* represents the effect of all possible specular reflections on distribution $I$. We can be sure that the infinite sum converges, as the eigenvalues of $\mathcal{T}$ have a module strictly less than one. Indeed, the energy balance within the enclosure states that the total reflected intensity is less than the incident intensity, the difference being absorbed by the various materials.



(a) Operator $\mathcal{T}$ (b) Operator $\mathcal{S}$ (c) Operator $\mathcal{D}$

Figure 2: Effects of the operators $\mathcal{T}$ (specular reflection-refraction), $\mathcal{S}$ (global specular reflection-refraction) and $\mathcal{D}$ (diffuse reflection) on a single light ray.

The distribution $\beta$ may be expressed in the same manner as :

$$\beta = \varepsilon + \mathcal{D} \cdot I. \qquad (2)$$

where $\mathcal{D}$ is the *diffuse reflection operator*, defined by :

$$\mathcal{D}I(x') = \rho^d(x') \int_S I(x', x'')dx''.$$

The operator $\mathcal{D}$ represents the effect of a single diffuse reflection (on all surfaces), on distribution $I$. It is the operator used in the conventional radiosity method.

Finally, replacing $I$ by its value given by equation (1) in equation (2), we find an equation on the isotropic distribution $\beta$.

$$\boxed{\beta = \varepsilon + \mathcal{D} \cdot \mathcal{S} \cdot g\beta.} \qquad (3)$$

### 3.2 How to use these equations

We shall now discuss how the above formulation of the rendering equation leads to a calculation algorithm. Until now, we only dealt with integral equations. By dividing the environment into patches of finite size, we can turn these integrals into summations over the

patches. Let us assume for now a purely specular behavior, which means that the function $\rho^s$ actually equals zero everywhere but in the exact reflected and refracted directions. This assumption is not really necessary here, we merely use it for sake of clarity, as it permits the use of a conventional ray-tracing algorithm.

The two passes of the algorithm will first estimate the isotropic distribution $\beta$, and then derive the complete distribution $I$, for the directions reaching the eye. It is important to see that none of these two passes can be omitted. The result of first pass is a distribution of light where each patch acts as a diffuse illuminator, even if the amount of energy emitted depends on the specular interactions within the environment (see figure 13, and comments in section 6). It should be stressed here that the discretization of the objects into patches is necessary only for the radiosity-like calculation, and not for the ray-tracing calculation. We can thus use a simpler, more compact representation of objects, to be used in all the ray-tracing part of the process.

### First pass : diffuse light.

Equation 3 gives us a way to calculate the "isotropic" distribution of light $\beta$. In fact, this is a radiosity equation, like the one introduced by Goral *et al.*, with the diffuse reflection operator being replaced by the product $\mathcal{D} \cdot \mathcal{S}$. The usual radiosity method solves this equation by computing geometrical form-factors, which represent the relationships between all patches in the scene. These form-factors are used to build the matrix of the diffuse reflection operator, and this matrix is then numerically inverted. This suggests that a radiosity method, in which only the form-factor calculation needs to be modified, will give us the distribution $\beta$. More precisely, the notion of form-factor will be extended to include specular effects.

The extended form-factors have a slightly different meaning, compared to the usual ones :

> $F_{ij}$ is the proportion of the energy leaving surface element $i$ and reaching surface element $j$, **after any number of specular reflections or refractions.**

Wallace *et al.* also use some extended form factors, but they only allow one reflection on planar mirrors. Our extended form factors are more general because they allow an arbitrary number of specular interactions, with patches of any geometry. The calculation of these extended form-factors can be derived just by closer examination of the equation. We want to model the action of the operator $\mathcal{D} \cdot \mathcal{S}$, which is equivalent to determining where the light received by some point – or surface element – comes from, after having been operated on by this operator. The operator $\mathcal{D}$ means that we must consider all the surface elements visible from that point, as in a classical form factor computation, and the operator $\mathcal{S}$ that, for each of these elements, we have to study a tree of reflected and refracted rays. The process is summarized in figure 3.

We can use these extended factors in a classical radiosity process : we form the matrix relating the distributions $\varepsilon$ and $\beta$ by multiplying the form-factors by the diffuse reflectance values for each wavelength band [8] , and we invert the matrix using an iterative Gauss-Seidel algorithm [9] (the actual matrix to be built and inverted is in fact the matrix of the operator $[1 - \mathcal{D}\mathcal{S}g]$, as $\beta = [1 - \mathcal{D}\mathcal{S}g]^{-1}\,\varepsilon$. It is worth noting here that the diffuse reflection coefficients can be changed very easily, as in the classical radiosity process, since they have no impact on the extended form-factors. Conversely, the specular coefficients are used in the computation of the extended form-factors, and thus can not be changed without re-calculating these factors.

```
For each surface element i {
   For each direction in space d {
      Trace a ray in direction d.
      Distribute the elementary form-factor of the
      direction among the objects in the ray-tracing tree.
   }
}
```

Figure 3: Calculation of the extended form-factors.

### Second pass : directional distribution

The directional distribution of light, $I$, must be computed for all directions of space converging to the observation point. We calculate this distribution with equation (1) derived above. The distribution $\beta$ has been calculated by the extended radiosity process, so that we just have to evaluate the effects of applying operator $\mathcal{S}$ to it. Let us recall that $\mathcal{S}$ represents the effect of any number of specular reflections–refractions. In order to compute $I$ for all directions reaching the eye, it is sufficient to use a classical ray-tracing algorithm from the eye position, with only the following modifications :

- No shadow rays are needed, which makes the process faster than conventional ray-tracing. Also the computation time is not dependent on the number of light sources.
- The "shading model" is trivial. It is simply the value of the distribution $\beta$ calculated at the given point.

### The general case for specular reflection

We modeled the global specular reflection operator as a *ray-tracing operator* because of the pure specular behavior assumption. It is possible to use a more complicated reflectance function, as long as there is a way to compute the effect of the operator $\mathcal{S}$. A distributed ray-tracing algorithm [4] could be used for this purpose, both in the computation of the extended form-factors, and in the second pass. Wallace's method, which uses z-buffer computations and simulates distributed ray tracing, could be used as well. It is important to note, however, that the same computational method should be used in the extended form factor computation (first pass), and in the final rendering (second pass), if true light transfer simulation is wanted. Otherwise we can obtain images with a very realistic rendering, including non-mirror specular reflectance functions, but with an incomplete "diffuse" solution, if pure ray tracing was used in the first pass.

## 3.3 Discussion

The two-pass approach is certainly a good compromise between image fidelity and CPU cost, as it uses the respective strength of ray tracing and radiosity to compute the different components of light. Although it involves the computation of extended form factors, which requires more CPU than the computation of diffuse form factors, this method has a number of advantages against Shao *et al.*'s method :

- It is independent of light distribution. If the extended form factors are stored in a file, a new picture can be generated with new lighting conditions, at the only expense of the radiosity solution and the second pass (see last section of this paper for a method to display the final images more quickly).
- There is no need to store the hemi-cubes for the specular patches.

- Specular patches do not have to be finely subdivided. A problem that appears when studying the reflection of light on a specular surface is that directions of interest vary rapidly on the surface (fig. 4). Therefore, all specular patches should be finely subdivided in order for Shao *et al.*'s method to produce accurate results.

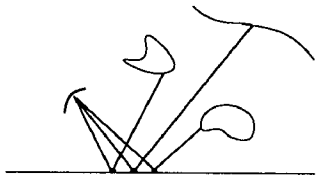


Figure 4: Precision in specular reflection.

On the other hand, Wallace's "image method" (though restricted to plane mirrors) and our ray tracing technique, allow specular patches to be treated as a whole, without further substructuring.

This lack of precision for specular reflection is precisely the reason that makes Shao *et al.* use a ray-traced second pass, and one can wonder if the precision of specular transfers is not as important during the first pass than during the second. Actually, it would make more sense to either use Shao's method as one pass, with finely subdivided specular patches (but then there will be a huge number of hemi-cubes to store), or a complete two-pass method like the one presented here.

However, Shao's method could prove very useful for simple environments, if a complex reflectance function is to be used.

## 4 An alternative to the *hemi-cube* sampling technique.

A new sampling method using adaptive subdivision is introduced for the calculation of form-factors, which is an alternative to the hemi-cube technique [3]. The use of an adaptive subdivision scheme proves especially useful when extended form-factors are to be computed, as it reduces the number of rays to be shot, but our statistics show that the approach is efficient even for the diffuse case, especially for high sampling resolutions. We shall first explain the method in the diffuse case, and then show how it can be used for extended form factors.

### 4.1 Adaptive sampling of the half-space

In a radiosity program, the form-factor calculation requires a sampling of the solid angle visible from a given surface element. We should therefore analyse a whole half-space above the tangent plane at this point, in order to account for all possible transfers of light. Cohen and Greenberg suggested to replace the sampling hemisphere by a hemi-cube of unit size, and to project the environment on its faces [3]. The faces are subdivided in "pixels" and a fast classical algorithm (z-buffer) is used to solve the visibility problems (figure 5). A surface element is thereby associated with each pixel, and the corresponding component of the form-factor (which has been pre-calculated) is added to the form-factor between the two elements. We present here a different method, where we project the environment only once, on a plane parallel to the tangent plane at the given point. We then choose to consider only a restricted area on this plane, thus neglecting all the portion of the half-space that projects itself out of the selected area. This approximation is

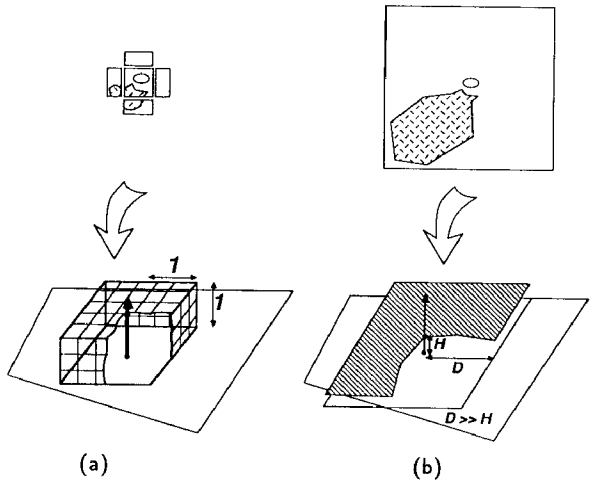



Figure 5: By using a single projection, the five images of the hemi-cube become a unique, very deformed image.

made possible by the angular dependency of the form-factor : in the calculation of the form-factor, one must integrate a numerical function depending on the cosine of the angle between the direction of sight and the surface normal. The contribution of directions that are nearly tangential to the surface considered is thus much smaller than the one of almost perpendicular directions. More precisely, the energy diffused through the differential cone shown on figure 6 is given by : $\triangle P = P_0 \sin 2\theta d\theta$ [8] ($P_0$ is the total radiated power).

We can estimate an upper bound for the energy fraction that is neglected when we analyse the energy being diffusely emitted through a square area of size $2D$, centered in a projection plane at a distance $H$ from the emitting surface (with $H \ll D$, see figure 6). Actually, the "lost" energy is less than the energy radiated in the directions with $\theta \in [\frac{\pi}{2} - \varepsilon, \frac{\pi}{2}]$, if $\tan \varepsilon = \frac{H}{D}$. The neglected energy fraction is such that :

$$\frac{\triangle P}{P_0} < \sin 2\varepsilon \cdot \varepsilon$$

$\varepsilon$ is small compared to one, so that we can write $\tan \varepsilon \approx \varepsilon$ and $\sin 2\varepsilon \approx 2\varepsilon$. Finally we get

$$\frac{\triangle P}{P_0} \approx 2\varepsilon^2 \approx 2(\frac{H}{D})^2$$

If, for example, we decide that an error of 1% is acceptable, we calculate the value of the ratio $\frac{D}{H}$ : $\frac{D}{H} \approx \varepsilon^{-1} \approx \sqrt{\frac{2P_0}{\triangle P}} \approx 14$. (We used this value in our implementation of the radiosity method). The above estimation relies on a lambertian distribution for the emitted light, but this condition is met even for the extended form factors, since light distribution $\beta$ is lambertian.

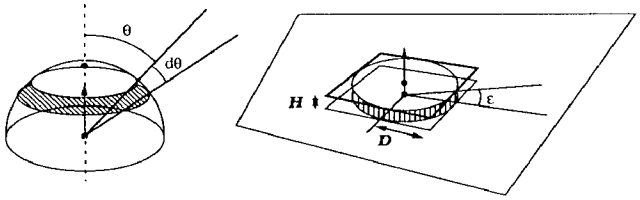



Figure 6: The differential solid angle is $d\omega = 2\pi \sin \theta d\theta$.

We now have to analyse the projection of the scene on our "screen". Due to the perspective distortion induced by the projection, it seems unreasonable to sample uniformly the inside of the square. This would lead to oversampling the external regions, in order to obtain a sufficient resolution at the center. We divide the screen in variable-size "proxels" (*pro*jection *el*ements), each proxel contributing for about the same amount to the form-factor.

The elementary form-factor associated from the origin with a rectangular area, bounded by $x_1, x_2, y_1, y_2$ (see figure 7) is given by [8][3] :

$$\triangle F_{x_1,x_2,y_1,y_2} = \int_{x_1}^{x_2} \int_{y_1}^{y_2} \frac{H^2}{(x^2+y^2+H^2)^2} dxdy.$$

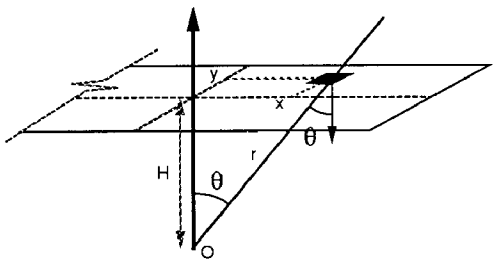



Figure 7: Geometry for the calculation of the elementary form-factors.

We want to find a sequence of integers $(x_i)_{i=0...N}$, $N$ being a fixed resolution, such that $x_0 = 0$, $x_N = D$, and

$$\forall i,j \quad \triangle F_{x_i,x_{i+1},x_j,x_{j+1}} \approx \frac{1}{N^2}.$$

Practically, due to the radial symmetry of the integrand, the above requirement is impossible to meet. We chose the values so that all $\triangle F$ along the axes have approximately the same value, by numerically estimating the integral (see figure 8).

We obtain a partition of the plane in rectangular regions, by a number of axis-parallel lines. The location of these lines (the $(x_i)_{i=0...N}$) needs only to be calculated once, for a given resolution. They are stored in a file, and will be used as proxel coordinates. For each patch in the scene, we want to analyse the projection of the environment on this rectangular grid, and associate another patch to each proxel.

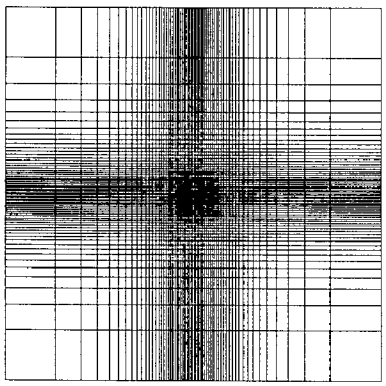


Figure 8: Subdivision of the screen in "proxels". We want to obtain regions with equal contributions to the form-factor.

In order to capitalize on the spatial coherence of the projection, we use an adaptive subdivision scheme to analyse the image on the screen, as introduced by Warnock [18].

The principle of this algorithm is to analyse the projected image in a rectangular region of the plane, or window. If the content of the window is "simple enough", or in other words, if the visibility problem is solved, the algorithm stops. In all other cases, the window is subdivided, and the process applied to the sub-windows.

Once the contents of a window have been identified as corresponding to a given surface element, we should add to the form-factor of this element the contribution of the window. The contribution of the different proxels, which are not all equal, are precalculated and stored in a table, but still, as a window could (and should) contain many proxels, we do not want to sum all the proxel contributions within the window. We therefore store in the table, in place of the $(i,j)$ proxel contribution, the sum of the contributions of all proxels $(p,q)_{p\leq i, q\leq j}$, following an idea used for example by Crow [6] to store textures. In this way, the contribution of any window can be estimated in constant time, with only one addition and two subtractions : if the integrals are stored in a bi-dimensional array $T$, we have :

$$\triangle F_{x_i,x_j,y_k,y_l} = T[j,l] - T[j,k] - T[i,l] + T[i,j].$$

### 4.2 Generation of the extended form factors

The above algorithm is able to associate an elementary form factor to any rectangular region of the projection plane, corresponding to diffuse emission towards this window. While for traditional radiosity the regions are simply patch visibility regions, this is no longer true for the extended form factors, since we want to follow the light along specular reflections. Actually, when Warnock's algorithm detects a window, and an associated patch, rays are shot at the corners of the window, and the ray trees are compared. A subdivision criterion is tested, which basically states that the first few levels of the tree should be the same. If this is not the case, the window is subdivided and new rays are shot. We see that the algorithm is only modified by a post-process to be executed for each Warnock window. Once a window is subdivided enough so that the ray-tracing trees at the corners match, the elementary form factor associated with the window is distributed among the objects in the tree. In order to avoid shooting the same rays several times during the subdivision, a storage algorithm has been developed, that only requires $2N$ storage locations, where $N^2$ is the number of proxels [7].

### 4.3 Comparison to the hemi-cube method

An analysis of running times has been performed with the two algorithms, for different sampling resolutions (figure 9). Times were obtained on a Bull DPX-5000 minicomputer, and represent the time needed to compute a whole set of diffuse form factors, for the scene shown on figure 15. The resolutions of the hemi-cube and the proxel plane can be compared because they correspond to the same sampling cell size at the center of the plane. However, both programs were software implementations. There is little doubt that a hardware-assisted hemi-cube, as suggested in [16], would be much faster.

We see that the two methods are comparable in time consumption, but that the hemi-cube times seem to increase more rapidly with the resolution. This is predictable, since a depth test must be performed at each pixel, while the subdivision in Warnock's algorithm depends mainly on the projected image. Asymptotically, for a given scene, the time needed by our algorithm is bounded by a linear growth, as the subdivision occurs only on "edges" in the

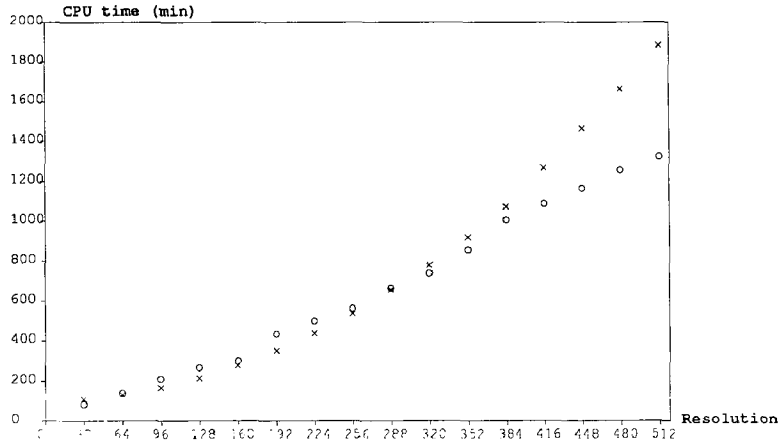



Figure 9: Comparison of running times (1152 patches). A resolution of $N$ means a $N \times N$ grid. (o) stand for our unique projection method, and (×) for the hemi-cube program.

image plane. Therefore, in the diffuse case, it can be interesting to use our method for large resolutions (which provide more accurate form factors).

When extended form factors are to be computed, for the extended two-pass method, our algorithm allows a strong decrease in the number of rays to be shot, and should therefore be preferred to the hemi-cube.

## 5 One step further towards interactivity

Cohen *et al.* [2] presented at Siggraph '88 a reformulation of the radiosity solution which allows the display of intermediate images, that gracefully converge towards the correct solution. This is a significant advance in the process of making realistic rendering practical for designers, because it makes the interaction loop shorter between the human and the machine. It is important to notice that the first pass of our extended two-pass method can be adapted in the same way. Our experiments show that for a "typical" scene such as the room shown on figure 15, progressively refined images integrating specular reflections and refractions of "diffuse" light are generated at a speed which is only 20 % less than that of Cohen *et al.*'s method (see table 1 and comments).

The problem remains, however, to interactively display a complete solution, including the eye-related specular effects (action of the operator $\mathcal{S}$). We present a method, based on hardware-assisted z-buffer and image composition, which solves this problem for planar mirrors. This method is not meant to replace the second pass completely, as it is limited to such mirrors, but instead the goal is to quickly display a picture incorporating some important specular effects. It should be stressed that the solution process as expressed by Cohen *et al.* is still independent of the viewpoint, so that a person sitting in front of a workstation is able to move through the scene while the solution progresses. We show how to accelerate the rendering of mirror effects when viewing conditions are allowed to change.

### 5.1 Quick generation of a picture with mirrors

If a particular patch is a planar mirror, one can view the intensity coming from a point on the patch towards the eye as a composition of the (diffuse) radiosity of the patch, and of the intensity arriving from the reflected direction. If a picture has been computed, with the viewpoint transferred to its reflected position relative to the mirror, one can easily retrieve this reflected intensity (figure 10).

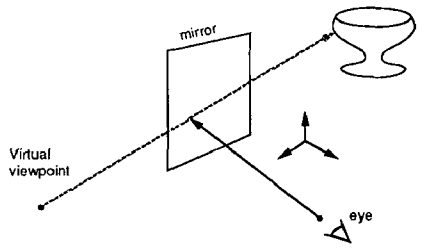



Figure 10: Calculation of a reflected image.

Care must be taken of a few points while computing the reflected image :

- Reflection on the mirror changes the orientation of the coordinate system.
- An additional clipping plane (the mirror itself) should be considered.

On most modern graphics workstation such as the Hewlett-Packard 835-SRX used in our implementation, an off-screen portion of the frame buffer can be used to compute the reflected image. Displaying the complete picture then only involves masking (to extract the portion of the picture where the mirror is seen) and a frame buffer to frame buffer copy of a block of pixels. Thus the extra time required to display the final picture, including first reflection on planar mirrors, is about the number of mirrors times the time needed for a z-buffered projection. For a simple scene like the one shown in figure 14, the display time is roughly doubled, which allows interaction with the picture (the complete display must be done each time the radiosity values are updated by the solution process).

### 5.2 Moving the viewpoint

When the viewpoint is to be moved "continuously", and for complex scenes, even a factor two in the display time is too much. On the other hand the precision required for displayed images is less, because each frame is displayed only for a very short time. To reduce the time spent for each picture, we actually store a reflected

picture bigger than the size of the mirror. If the viewpoint is far enough from the mirror for the perspective to be almost parallel, the projected image for a new viewpoint can be approximated by a sub-image of this current reflected image. The additional z-buffer calculation can thus be performed less often, say for example each third or fourth picture.

Figure 11 explains how the "old" reflected image is used to create a new mirror reflection, when the view point moves in a plane perpendicular to the mirror (on figure 11 the mirror is "vertical" and the viewpoint moves in a horizontal plane). Index 1 denotes the "old" viewing situation, and index 2 the new one, for which the picture must be calculated. The idea is to use the portion of the old image (segment) indicated by the thick line. Lengths $L_1$ and $L_2$ are easily calculated given the viewpoints and the mirror, and represent the widths of the mirror images (they can be expressed in pixels). The main problem here is to choose a "distance" parameter (called $h$ in the figure) that will be used to correlate the two views. Small values of $h$ would result in a negligible difference between the pictures, while large values would destroy all coherence between them.

Noting that

$$\frac{u}{v} = 1 + \frac{L_1}{h \tan \beta_1} \qquad \text{and} \qquad L_1 + u = L_2 + v \quad ,$$

we use a fixed $h$ value, and calculate $u$ and $v$ accordingly. These values represent the displacement of the reflected picture in the image plane. Generalization is straightforward, and we see that an approximate reflected picture can be generated with very little computation.

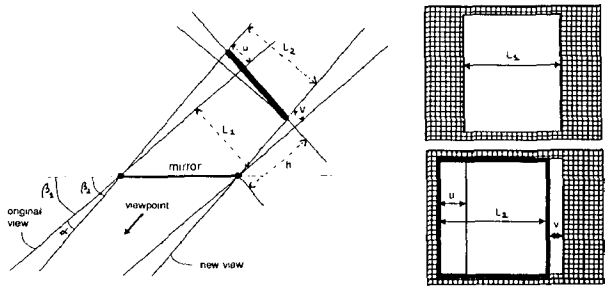



Figure 11: Estimating the translation in image plane.

## 6 Results

### A transparent sphere

Figure 12 shows different renderings of a simple test scene, composed of a transparent sphere illuminated by three colored spotlights. The effect of treating refraction during the first pass is particularly visible.

### Why two passes ?

Figure 13 shows different treatments of the same scene, calculated under the same lighting conditions. It is apparent that the inclusion of specular effects is critical to the realism of the image (Illumination of the table and of the back of the vase for example).

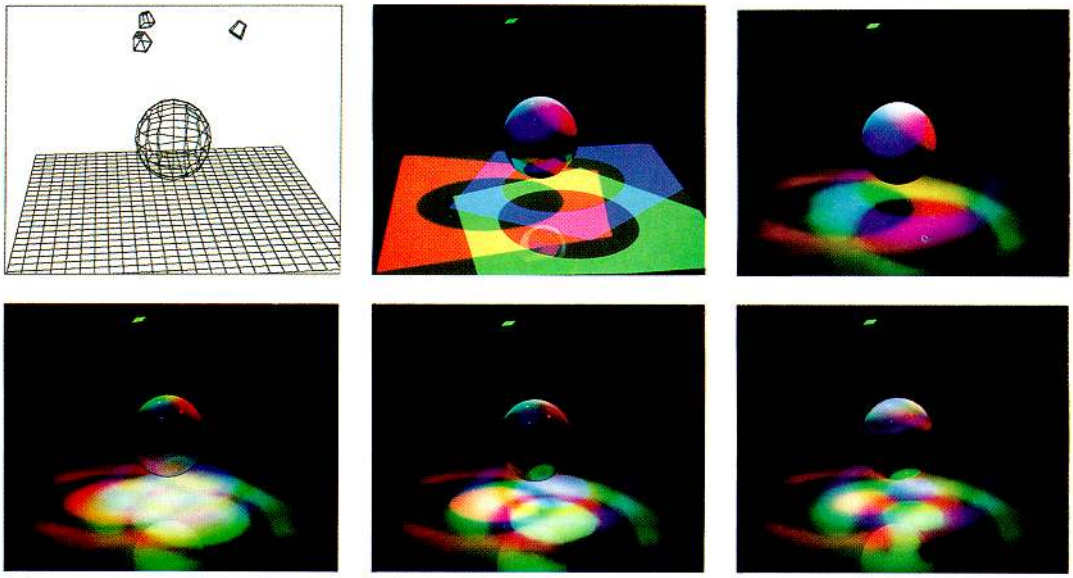


Figure 12: Top to bottom, left to right : (1) Scene geometry. (2) Ray-traced picture. (3) Conventional Radiosity. (4) Refraction index 1.025. (5) Refraction index 1.1. (6) Refraction index 2.0.

Note that the complex colored shadows on the floor are rendered by ray-tracing, but the shadow boundaries are too sharp. Furthermore the light traveling from each spotlight, through the sphere, and reaching the ground is ignored. The general aspect of the radiosity picture is good, with penumbrae effects, but some lighting effects are still ignored. The output of our general system is presented for three different values of the refracting index of the sphere. The realism of these pictures is due in a large part to the light refracted by the sphere on the plane. We can see that, as the refraction index increases, the illuminated area in the geometrical shadow of the sphere first shrinks, and then grows and becomes more diffuse. This is due to the modification of the light's path through the sphere with the refraction index. Similarly, we see that this illuminated area takes the color of the corresponding spotlight when it is concentrated enough. This area is actually illuminated by the three spotlights, but receives more light through the sphere, because of the concentration of rays.

Figure 13: (a) Diffuse solution. (b) Diffuse first pass, with a ray-traced second pass. (c) Complete two-pass solution.

Figure 14: (a) Virtual image. (b) Approximation for translated viewpoint. (c) Virtual image, for translated viewpoint.

### Mirror simulation

Figure 14 shows different views from the room shown on figure 15, obtained with the composition method explained in section 5. The first picture (a) was produced by computing the reflected image from a virtual viewpoint. The second picture (b) illustrates the use of a simple translation in the relected image plane, when the viewpoint is moved to the right. Note that the light on the lower table becomes visible in the mirror. The third image (c) uses a virtual viewpoint again, but with the new viewing conditions, for comparison with the approximation in (b). We can see that the image of the light on the lower table is correctly placed by the approximation, while the light closer to the mirror was moved too far to the right. This is an illustration of the "correlation distance" $h$, which means that only objects within a certain distance range from the mirror will be correctly rendered.

Figure 15: The room used for the mirror simulation of figure 14

### Computation times

Calculation of a complete row of form factors (seconds).

| | Number of patches | Normal form factors | Extended form factors |
|---|---|---|---|
| room (fig. 15) | 1152 | 12.18 | 14.41 |
| sphere (fig. 12) | 802 | 3.60 | 16.2 |

Second pass (1280 × 1024 pixels, in minutes).

| | Second pass | Conventional ray tracing |
|---|---|---|
| room | 108 | 196 |
| sphere | 76 | 148 |

Table 1: Computation times (Bull DPX 5000).

Some computation times are given in table 1. They indicate the average time spent calculating a row of form factors, for two test scenes. This parameter was chosen because it gives the radiosity refresh rate in the progressive refinement program. However our implementation is distributed over a network of workstations, and this refresh time is actually divided by the number of processors. The extended form-factors are of course more expensive to compute, but for environments containing few specular patches, we see that the ratio remains very reasonable. The ray tracing times listed show that the second pass is appreciably faster than a complete conventional ray tracing, because no shadow rays are traced.

## 7 Conclusions

A reformulation of previous global interreflection algorithms within the framework of the *rendering equation* has been presented. This formulation uses *reflection operators* to describe the interaction of light with the objects, and allows a more precise comparison of different algorithms. This presentation shows the power of a two-

pass approach, as compared to Shao *et al.*'s progressive refinement method.

Our general two-pass method extends the work of Wallace *et al.*, by allowing multiple specular reflections and refractions, and by removing the previous restriction to planar mirrors. This generalization is very important for promoting a broader use of such light simulation methods.

A new sampling method for the calculation of form factors has been presented here; it is an alternative to the hemi-cube technique. The method capitalizes on the coherence shown by an environment's view, taken from a random patch. Analysis shows that the method is comparable to the hemi-cube technique for classical radiosity, and may be even better as the resolution is increased. However, it proves especially interesting when ray-tracing is used in the calculation of extended form-factors, because it provides a way to reduce the number of rays being traced.

Finally, the mirror simulation described allows the construction of an interactive system that effectively simulates simple specular surfaces. This method should be used together with the progressive refinement technique of Cohen *et al.*, and could for example be the basis of an architectural simulation software.

Future work on interactive display of complete pictures will include efficient ways to cope with rotations of the view direction, without having to project the whole scene for each view. A preprocessing could for example associate to "each" direction the set of objects which can be seen from a given mirror in that direction.

Some further directions deserve investigation: first, no account is taken of the possible interaction of light with the propagation medium. The zonal method, presented by Rushmeier and Torrance [14], could probably be integrated with the system, for it is based on a radiosity approach. Some other approximations on the reflectance functions should also be tried out, together with the associated form-factor computation algorithms. As mentioned above, a distributed ray-tracing algorithm could be readily used in order to take into account a more sophisticated reflectance function. But better algorithms could certainly be used in place of this brute force method.

The adaptation of the method to parallel or distributed computers also deserves attention, for the main part of the computation time is devoted to geometric calculations that are largely independent of one another.

Figure 16: A test scene for the two-pass method. Hevea tree database courtesy of AMAP.

## Acknowledgements

Particular thanks go to Olivier Devillers, who helped clarify the ideas presented here, and wrote the ray-tracing code used in the program. We would also like to thank Michael Cohen for his very helpful comments on an early version of this paper.

## References

[1] James F. Blinn. Models of light reflection for computer synthesized pictures. *Computer Graphics*, 11:192–198, 1977. Proceedings SIGGRAPH 1977.

[2] Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg. A progressive refinement approach to fast radiosity image generation. *Computer Graphics*, 22(4):75–84, August 1988. Proceedings SIGGRAPH 1988 in Atlanta.

[3] Michael F. Cohen and Donald P. Greenberg. The hemi-cube : A radiosity solution for complex environments. *Computer Graphics*, 19(3):31–40, July 1985. Proceedings SIGGRAPH 1985 in San Fransisco.

[4] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *Computer Graphics*, 18:137–147, July 1984. Proceedings SIGGRAPH 1984 in Minneapolis.

[5] Robert L. Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics*, 1:7–24, 1982.

[6] Franklin C. Crow. Summed-area tables for texture mapping. *Computer Graphics*, 18:207–212, July 1984. Proceedings SIGGRAPH 1984 in Minneapolis.

[7] Olivier Devillers, Claude Puech, and François Sillion. *CIL : un modèle d'illumination intégrant les réflexions diffuse et spéculaire.* Technical Report 87-12, Laboratoire d'Informatique de l'ENS, 45 rue d'Ulm, 75230 Paris Cedex 05, France, October 1987.

[8] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. *Computer Graphics*, 18(3):213–222, July 1984. Proceedings SIGGRAPH 1984 in Minneapolis.

[9] Robert W. Hornbeck. *Numerical Methods.* Quantum Publishers, 1975.

[10] David S. Immel, Michael F. Cohen, and Donald P. Greenberg. A radiosity method for non-diffuse environments. *Computer Graphics*, 20(4):133–142, August 1986. Proceedings SIGGRAPH 1986 in Dallas.

[11] James T. Kajiya. The rendering equation. *Computer Graphics*, 20(4):143–150, August 1986. Proceedings SIGGRAPH 1986 in Dallas.

[12] T. Nishita and E. Nakamae. Continuous tone representation of three-dimesional objects taking account of shadows and interreflection. *Computer Graphics*, 19(3):23–30, July 1985. Proceedings SIGGRAPH 1985 in San Francisco.

[13] Bui-Tuong Phong. *Illumination for Computer Generated Images.* PhD thesis, University of Utah, 1973.

[14] Holly E. Rushmeier and Kenneth E. Torrance. The zonal method for calculating light intensities in the presence of a participating medium. *Computer Graphics*, 21(4):293–302, July 1987. Proceedings SIGGRAPH 1987 in Anaheim.

[15] Min-Zhi Shao, Qun-Sheng Peng, and You-Dong Liang. A new radiosity approach by procedural refinements for realistic image synthesis. *Computer Graphics*, 22(4):93–101, August 1988. Proceedings SIGGRAPH 1988 in Atlanta.

[16] François Sillion. *Simulation de l'éclairage pour la synthèse d'images : Réalisme et Interactivité.* PhD thesis, Université Paris XI, June 1989. (available from LIENS).

[17] John R. Wallace, Michael F. Cohen, and Donald P. Greenberg. A two-pass solution to the rendering equation : a synthesis of ray-tracing and radiosity methods. *Computer Graphics*, 21(4):311–320, July 1987. Proceedings SIGGRAPH 1987 in Anaheim.

[18] Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. *Computer Graphics*, 22(4):85–92, August 1988. Proceedings SIGGRAPH 1988 in Atlanta.

[19] John E. Warnock. *A Hidden-Surface Algorithm for Computer Generated Halftone Pictures.* Technical Report 4-15, University of Utah Computer Science Dept., June 1969. NTIS AD 753 671.