EUROGRAPHICS '99 / P. Brunet and R. Scopigno
(Guest Editors)

Volume 18 (*1999*), *Number 3*

Stochastic Iteration for Non-diffuse Global Illumination

László Szirmay-Kalos

Department of Control Engineering and Information Technology, Technical University of Budapest
Budapest, Műegyetem rkp. 11, H-1111, HUNGARY
szirmay@fsz.bme.hu

Abstract

This paper presents a single-pass, view-dependent method to solve the rendering equation, using a stochastic iterational scheme where the transport operator is selected randomly in each iteration. The requirements of convergence are given for the general case. To demonstrate the basic idea, a very simple, continuous random transport operator is examined, which gives back the light tracing algorithm incorporating Russian roulette. Then, a new mixed continuous and finite-element based iteration method is proposed, which uses ray-bundles to transfer the radiance in a single random direction. The resulting algorithm is fast, it provides initial results in seconds and accurate solutions in minutes and does not suffer from the error accumulation problem and the high memory demand of other finite-element and hierarchical approaches.

Keywords: Rendering equation, global radiance, Monte-Carlo integration, light-tracing, global ray-bundle tracing.

1. Introduction

Global illumination algorithms aim at obtaining the power detected by a collection of measuring devices. The measurement process is characterized by the following equation

$$\int_S \int_\Omega L(\vec{y},\omega)\cdot\cos\theta\cdot W^e(\vec{y},\omega)\,d\vec{y}\,d\omega = \mathcal{M}L, \qquad (1)$$

where $L(\vec{y},\omega)$ is the *radiance*, θ is the angle between the surface normal and direction ω and $W^e(\vec{y},\omega)$ is the *sensitivity* of the measuring device. A measuring device can detect, for example, the power leaving a set of points in a set of directions or the power reaching the eye through pixel P.

The evaluation of the detected power requires the radiance at points where the measuring device is focused to. The radiance function can be obtained by solving the *rendering equation*[14] that has the following form:

$$L = L^e + \mathcal{T}L. \qquad (2)$$

In this integral equation, operator $\mathcal{T}$ describes the light transport

$$\mathcal{T}L(\vec{x},\omega) = \int_\Omega L(h(\vec{x},-\omega'),\omega')\cdot f_r(\omega',\vec{x},\omega)\cdot\cos\theta'\,d\omega' \qquad (3)$$

where $L(\vec{x},\omega)$ and $L^e(\vec{x},\omega)$ are the radiance and emission of the surface in point $\vec{x}$ at direction ω, Ω is the directional sphere, $h(\vec{x},\omega')$ is the visibility function defining the point that is visible from point $\vec{x}$ at direction ω', $f_r(\omega',\vec{x},\omega)$ is the bi-directional reflection/refraction function, and θ' is the angle between the surface normal and direction $-\omega'$ (figure 1).

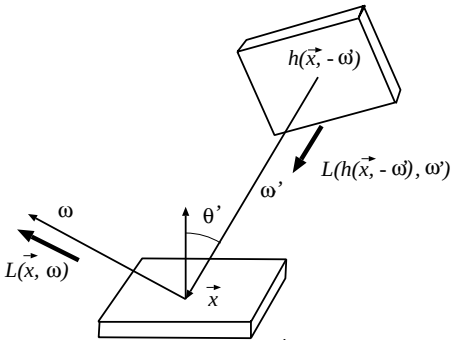

Figure 1: *Geometry of the rendering equation*

Since the rendering equation contains the unknown radi-

ance function both inside and outside the integral, in order to express the solution, this coupling should be resolved. The possible solution techniques fall into one of the following three categories: inversion, expansion or iteration.

1.1. Inversion

Inversion groups the terms that contain the unknown function on the same side of the equation and applies formally an inversion operation:

$$(1-\mathcal{T})L = L^e \implies L = (1-\mathcal{T})^{-1}L^e. \qquad (4)$$

Thus the measured power is

$$\mathcal{M}L = \mathcal{M}(1-\mathcal{T})^{-1}L^e. \qquad (5)$$

However, since $\mathcal{T}$ is infinite dimensional, it cannot be inverted in closed form. Thus it should be approximated by a finite dimensional mapping, that is usually given as a matrix. The inversion is then the solution of the resulting linear equation.

1.2. Expansion

Expansion substitutes function L in the right side by the complete right side (which equals to L) recursively. If the integral operator is a contraction, this provides the solution in the form of an infinite Neumann series:

$$L = L^e + \mathcal{T}L = L^e + \mathcal{T}(L^e + \mathcal{T}L) = L^e + \mathcal{T}L^e + \mathcal{T}^2L =$$

$$L^e + \mathcal{T}L^e + \mathcal{T}^2(L^e + \mathcal{T}L) = \ldots = \sum_{i=0}^{\infty} \mathcal{T}^i L^e. \qquad (6)$$

Thus the measured power is

$$\mathcal{M}L = \sum_{i=0}^{\infty} \mathcal{M}\mathcal{T}^i L^e. \qquad (7)$$

The main problem of expansion techniques is that they require the evaluation of very high dimensional integrals that appear as terms in the infinite series. In order to avoid dimensional explosion — i.e. evaluating a D-dimensional integral with $O(M^D)$ time complexity — Monte-Carlo or quasi-Monte Carlo techniques should be used.

On the other hand, expansion techniques also have an important advantage. Namely, they do not use temporary representations of the complete radiance function, thus do not necessitate finite-element approximations. Consequently, these algorithms can work with the original geometry without tessellating the surfaces to planar polygons.

In computer graphics the first Monte-Carlo random walk algorithm — called *distributed ray-tracing* — was proposed by Cook et al. [9], which spawned to a set of variations, including *path tracing*[14], *light-tracing*[11], *bi-directional path tracing*[16, 42], *Monte-Carlo radiosity*[30, 18, 23], and *two-pass methods* which combine radiosity and ray-tracing [29, 45, 44].

To reduce the variance of the Monte-Carlo integration, most of these methods incorporate some form of the *importance sampling*[34]. The importance can be based on the local BRDFs [11, 16], on the direct illumination [31], on both [41, 17], or can even be explored adaptively [43].

Expansion techniques generate random walks independently. It can be an advantage, since these algorithms can be parallelized easily. However, it also means that these methods "forget" the previous history of walks and cannot reuse the visibility information gathered in previous walks, thus they are not as fast as they could be.

Some sort of reusing the visibility information for many walks happens, for example, in the *multi-path methods*. They are essentially random walk methods, but in their single step many random walks are advanced parallely. Sbert[25, 27, 26] proposed a complete family of multi-path methods, that are based on random global lines, which is the basic "engine" to advance the walks. A single global line transfers the reflected power of all those patches that are intersected by this line to the direction of this line. The global line also transfers a portion of the emission of the intersection patches. Thus a line initiates those walks that would start in a patch intersected by this line, and continues those previous walks which carried some power onto the intersected patches.

1.3. Iteration

Iterational techniques realize that the solution of integral equation (2) is the fixed point of the following iterational scheme

$$L_n = L^e + \mathcal{T}L_{n-1}, \qquad (8)$$

thus if operator $\mathcal{T}$ is a contraction, then this scheme will converge to the solution from any initial function L_0.

The measured power can be obtained as a limiting value

$$\mathcal{M}L = \lim_{n\to\infty} \mathcal{M}L_n, \qquad (9)$$

In order to store the approximating functions L_n, usually finite-element techniques are applied, as for example, in diffuse radiosity[8], or in non-diffuse radiosity using partitioned hemisphere[13], directional distributions[32] or illumination networks[4].

Compared to expansion techniques, iteration has both advantages and disadvantages. Its important advantage is that it can potentially reuse all the information gained in previous computation steps, thus iteration is expected to be faster than expansion. Iteration can also be seen as a single infinite length random walk. If implemented carefully, iteration does not reduce the number of estimates for higher order interreflections, thus it is more robust when rendering highly reflective scenes than expansion.

The property that iteration requires tessellation and finite-element representation is usually considered as a disadvantage. And indeed, sharp shadows and highlights on highly

specular materials can be incorrectly rendered and light-leaks may appear, not to mention the unnecessary increase of the complexity of the scene description (think about, for example, the definition of the original and tessellated sphere). However, finite-element representation can also provide smoothing during all stages of rendering, which results in more visually pleasing and dot-noise free images.

Iteration has two critical problems. On the one hand, since the domain of L_n is 4 dimensional, an accurate finite-element approximation usually requires very many basis functions, which, in turn, need a lot of storage space. Although, *hierarchical methods*[12, 2], *wavelet* or *multiresolution methods*[7, 28] and *clustering*[33, 6, 35] can help, the memory requirements are still prohibitive for complex scenes. On the other hand, when finite element techniques are applied, operator $\mathcal{T}$ is only approximated, which introduces some non-negligible error in each step. If the contraction ratio of the operator is λ, then the total accumulated error will be approximately $1/(1-\lambda)$ times the error of a single step[39], which can be unacceptable for highly reflective scenes. For highly reflective scenes, the iteration is slow and the result is inaccurate if the approximation of the operator is not very precise. Very accurate approximations of the transport operator, however, require a lot of computation time and storage space.

Both the problem of prohibitive memory requirements and the problem of error accumulation can be successfully attacked by *stochastic iteration.*

2. Stochastic iteration

The basic idea of stochastic iteration is that instead of approximating operator $\mathcal{T}$ in a deterministic way, a much simpler random operator is used during the iteration which "behaves" as the real operator just in the "average" case.

The concept of stochastic iteration has been proposed and applied for the diffuse radiosity problem in [18, 19, 21, 39], that is for the solution of finite-dimensional linear equations. In this section we generalize the fundamental concepts to solve integral equations, then the generalized method will be used for attacking non-diffuse global illumination problems.

Suppose that we have a random linear operator $\mathcal{T}^*$ so that

$$E[\mathcal{T}^* L] = \mathcal{T} L \tag{10}$$

for any integrable function L. This requirement can be explained as follows. Having applied the random operator $\mathcal{T}^*$ to a radiance function $L(p)$ (p denotes both the position $\vec{x}$ and the direction ω), we obtain a random variable which is required to have an expected value that is equal to $(\mathcal{T}L)(p)$ for every p.

During stochastic iteration a random sequence of operators $\mathcal{T}_1^*, \mathcal{T}_2^*, \ldots \mathcal{T}_i^* \ldots$ is generated, which are instantiations of $\mathcal{T}^*$, and this sequence is applied to the radiance function:

$$L_n = L^e + \mathcal{T}_n^* L_{n-1}. \tag{11}$$

Since in computer implementations the calculation of a random operator may invoke finite number of random number generator calls, we are particularly interested in those random operators which have the following construction scheme:

1. Random "point" p_i is found from a finite dimensional set Π using probability density $\text{prob}(p)$. This probability density may or may not depend on function L.
2. Using p_i a "deterministic" operator $\mathcal{T}^*(p_i)$ is applied to radiance L.

Point p_i is called the *randomization point* since it is responsible for the random nature of operator $\mathcal{T}^*$.

Using a sequence of random transport operators, the measured power

$$P_n = \mathcal{M} L_n \tag{12}$$

will also be a random variable which does not converge but fluctuates around the real solution. Thus the solution can be found by averaging the estimates of the subsequent iterational steps.

Formally the sequence of the iteration is the following:

$$\begin{aligned}
P_1 &= \mathcal{M}L_1 = \mathcal{M}(L^e + \mathcal{T}_1^* L^e), \\
P_2 &= \mathcal{M}L_2 = \mathcal{M}(L^e + \mathcal{T}_2^* L^e + \mathcal{T}_2^* \mathcal{T}_1^* L^e), \\
&\vdots \\
P_M &= \mathcal{M}L_M = \mathcal{M}(L^e + \mathcal{T}_M^* L^e + \mathcal{T}_M^* \mathcal{T}_{M-1}^* L^e + \ldots).
\end{aligned}$$

Averaging the first M steps, we obtain:

$$\tilde{P} = \frac{1}{M}\sum_{i=1}^{M} \mathcal{M}L_i =$$

$$\mathcal{M}(L^e + \frac{1}{M}\sum_{i=1}^{M} \mathcal{T}_i^* L^e + \frac{1}{M}\sum_{i=1}^{M-1} \mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e + \ldots) =$$

$$\mathcal{M}(L^e + \frac{1}{M}\sum_{i=1}^{M} \mathcal{T}_i^* L^e + \frac{M-1}{M} \cdot \frac{1}{M-1}\sum_{i=1}^{M-1} \mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e + \ldots). \tag{13}$$

In order to prove that $\tilde{P}$ really converges to the solution of the integral equation, first it is shown that the expectation value of

$$\mathcal{T}_{i+k}^* \mathcal{T}_{i+k-1}^* \ldots \mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e$$

is $\mathcal{T}^{k+1} L^e$. For $k = 0$, it comes directly from the requirement of equation (10). For $k = 1$, the *total expected value theorem*[24] can be applied:

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e] = \int_{\Pi} E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] \cdot \text{prob}(p)\, dp. \tag{14}$$

Since for a fixed $p_{i+1} = p$, operator $\mathcal{T}_{i+1}^*$ becomes a deterministic linear operator, its order can be exchanged with that of the expected value operator:

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] = \mathcal{T}_{i+1}^*(p) \left(E[\mathcal{T}_i^* L^e]\right). \qquad (15)$$

Using requirement (10) for the expected value we further obtain

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] = \mathcal{T}_{i+1}^*(p)(\mathcal{T} L^e). \qquad (16)$$

Substituting this back to equation (14), we get

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e] = \int_\Pi \mathcal{T}_{i+1}^*(p)(\mathcal{T} L^e) \cdot \mathrm{prob}(p)\, dp =$$

$$E[\mathcal{T}_{i+1}^*(\mathcal{T} L^e)] = \mathcal{T}(\mathcal{T} L^e) = \mathcal{T}^2 L^e. \qquad (17)$$

which concludes our proof for the $k = 1$ case. The very same idea can be used recursively for more than two terms.

Returning to the averaged solution $\tilde{P}$, its expected value $E[\tilde{P}]$ is then

$$\mathcal{M}(L^e + \mathcal{T} L^e + \frac{M-1}{M}\mathcal{T}^2 L^e + \frac{M-2}{M}\mathcal{T}^3 L^e + \ldots + \frac{1}{M}\mathcal{T}^M L^e),$$

which converges to the real solution

$$\mathcal{M}(L^e + \mathcal{T} L^e + \mathcal{T}^2 L^e + \mathcal{T}^3 L^e + \ldots)$$

if M goes to infinity.

Note also that there is some power "defect" because of the missing higher order terms for finite M values. Denoting the contraction ratio of the integral operator $\mathcal{T}$ by λ, and assuming that the measuring device is calibrated to show unit power for unit homogeneous radiance, this defect can be estimated as follows [37]:

$$|\Delta P| \leq \frac{1}{M} \cdot \frac{\lambda^2}{(1-\lambda)^2} \cdot ||L^e||. \qquad (18)$$

This can be neglected for high number of iterations, or can even be reduced by ignoring the first few iterations in the averaged result [18, 25].

Finally, it must be explained why random variable $\tilde{P}$ converges to its expected value. Looking at formula (13) we can realize that it consists of sums of the following form:

$$\frac{1}{M-k} \cdot \sum_{i=1}^{M-k} \mathcal{T}_{i+k}^* \mathcal{T}_{i+k-1}^* \ldots \mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e.$$

According to the theorems of large numbers, and particularly to the Bernstein[24] theorem, these averages really converge to the expected value if the terms in the average are not highly correlated (note that here the terms are not statistically independent as assumed by most of the laws of large numbers). It means that random variables $\mathcal{T}_{i+k}^* \mathcal{T}_{i+k-1}^* \ldots \mathcal{T}_i^* L^e$ and $\mathcal{T}_{j+k}^* \mathcal{T}_{j+k-1}^* \ldots \mathcal{T}_j^* L^e$ should not have strong correlation if $i \neq j$. This is always true if the sequence of operators are generated from independent random variables, which will be the case in the proposed algorithm.

3. Definition of the random transport operator

In order to use this general stochastic iterational scheme in practice, the key problem is the definition of the random transport operator. This operator should meet the requirement of equation (10), should be easy to compute and it should allow the compact representation of the $\mathcal{T}_i^* L$ functions. Generally the domain of L is a 4-dimensional continuous space, so is the domain of $\mathcal{T}_i^* L$ (for ray-bundle tracing only 2-dimensional continuous space). Considering the requirement of compact representation, we have to avoid the representation of these functions over the complete domain.

Thus those transport operators are preferred, which require the value of L just in a few "domain points" (e.g. in a single "domain point"). Note that the evaluation of $\mathcal{T}_i^* L$ now consists of the following steps: first a randomization point p_i is found to define random operator $\mathcal{T}_i^*$, which in turn determines at which domain point the value of L is required. Up to now, we have had complete freedom to define the set of randomization points. One straightforward way is defining this set to be the same as (or a superset of) the domain of the radiance function and using random transport operators that require the value of the radiance function at their randomization points. Although this equivalence is not obligatory, it can significantly simplify the computations, since when the randomization point is generated, the required domain point is also known.

Using random operators that evaluate the radiance in a single point is not enough in itself, since even a single "point" can result in a continuous $\mathcal{T}_i^* L$ function, which must be stored and re-sampled in the subsequent iteration step and also by the measurement. The solution is the postponing of the complete calculation of $\mathcal{T}_i^* L$ until it is known where its value is needed in the next iteration step and by the measuring device. In this way, the random operator should be evaluated twice but just for two points. Once for the actual and the previous "points" resulting in $[\mathcal{T}^*(p_i)L(p_i)](p_{i+1})$, and once for p_{eye} which is needed by the measuring device and for previous point, providing $[\mathcal{T}^*(p_i)L(p_i)](p_{\mathrm{eye}})$.

The complete iteration goes as follows:

$P = 0$
Find p_1 randomly
$L(p_1) = L^e(p_1)$
for $i = 1$ **to** M **do**
  $P^{\mathrm{new}} = \mathcal{M}(L^e(p_{\mathrm{eye}}) + [\mathcal{T}^*(p_i)L(p_i)](p_{\mathrm{eye}}))$
  $P = P^{\mathrm{new}} \cdot 1/i + (1 - 1/i) \cdot P$
  Find p_{i+1} randomly
  $L(p_{i+1}) = L^e(p_{i+1}) + [\mathcal{T}^*(p_i)L(p_i)](p_{i+1})$
endfor
Display final image

In the following sections two stochastic iteration schemes are considered from the many possible alternatives. The first alternative serves solely demonstration purposes and gives back the well-known light-tracing algorithm incor-

porating Russian roulette. The second alternative is a new and more effective version of the global ray-bundle tracing algorithm[40, 36].

4. Single ray based transport operator

In the method presented in this section, the random transport operator uses a single ray having random origin $\vec{y}_i$ and direction ω_i generated with a probability that is proportional to the cosine weighted radiance of this point at the given direction. This ray transports the whole power

$$\Phi = \int_S \int_\Omega L(\vec{y}, \omega') \cos\theta_{\vec{y}} \, d\omega' \, d\vec{y}$$

to that point $\vec{x}$ which is hit by the ray. Formally, the random transport operator is

$$(\mathcal{T}^* L)(\vec{x}, \omega) = \Phi \cdot \delta(\vec{x} - h(\vec{y}, \omega_i)) \cdot f_r(\omega_i, \vec{x}, \omega). \qquad (19)$$

Let us prove that this random operator meets the requirement of equation (10). The probability density of selecting surface point $\vec{y}$ and direction ω' is

$$\frac{d\Pr\{\vec{y}, \omega'\}}{d\vec{y}\, d\omega_{\vec{y}}} = \frac{L(\vec{y}, \omega') \cdot \cos\theta_{\vec{y}}}{\Phi} \qquad (20)$$

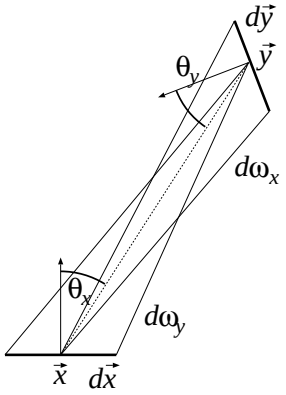

Figure 2: *Symmetry of solid angles of shooting and gathering*

Using the definition of the solid angle

$$d\omega_{\vec{y}} = \frac{d\vec{x} \cdot \cos\theta'_{\vec{x}}}{||\vec{y} - \vec{x}||^2}$$

we can obtain a symmetry relation (figure 2) for the shooting and gathering solid angles:

$$d\vec{y} \cdot d\omega_{\vec{y}} \cdot \cos\theta_{\vec{y}} = d\vec{y} \cdot \frac{d\vec{x} \cdot \cos\theta'_{\vec{x}}}{||\vec{y} - \vec{x}||^2} \cdot \cos\theta_{\vec{y}} =$$

$$d\vec{x} \cdot \frac{d\vec{y} \cdot \cos\theta_{\vec{y}}}{||\vec{y} - \vec{x}||^2} \cdot \cos\theta'_{\vec{x}} = d\vec{x} \cdot d\omega'_{\vec{x}} \cdot \cos\theta'_{\vec{x}}. \qquad (21)$$

Thus the probability of selecting $\vec{y}, \omega'$ can also be expressed in the following way:

$$d\Pr\{\vec{y}, \omega'\} = \frac{L(\vec{y}, \omega') \cdot \cos\theta_{\vec{y}}}{\Phi} \cdot d\vec{y}\, d\omega_{\vec{y}} =$$

$$\frac{L(h(\vec{x}, -\omega'), \omega') \cdot \cos\theta_{\vec{x}}}{\Phi} \cdot d\vec{x}\, d\omega'_{\vec{x}}. \qquad (22)$$

Now we can easily prove that the random transport operator meets requirement (10) since

$$E[(\mathcal{T}^* L)(\vec{x}, \omega)] =$$

$$\int_S \int_\Omega \Phi \cdot \delta(\vec{x} - h(\vec{y}, \omega')) \cdot f_r(\omega', \vec{x}, \omega) \, d\Pr\{\vec{y}, \omega'\} =$$

$$\int_\Omega L(h(\vec{x}, -\omega'), \omega') \cdot \cos\theta'_{\vec{x}} \cdot f_r(\omega', \vec{x}, \omega) \, d\omega'_{\vec{x}} = (\mathcal{T} L)(\vec{x}, \omega).$$

Note that the result of the application of the random operator can be a single point that receives all the power and reflects some part of it or the result can be no point at all if the ray leaves the scene.

Suppose that the first random operator $\mathcal{T}_1^*$ is applied to L^e which may transfer all power

$$\Phi_1 = \int_S \int_\Omega L^e(\vec{y}_1, \omega_1) \cos\theta_{\vec{y}_1} \, d\omega_1 \, d\vec{y}_1,$$

to a single point $\vec{x}_1 = h(\vec{y}_1, \omega_1)$ using probability density

$$\frac{d\Pr_1\{\vec{y}_1, \omega_1\}}{d\vec{y}_1 d\omega_1} = \frac{L^e(\vec{y}_1, \omega_1) \cdot \cos\theta_{\vec{y}_1}}{\Phi}.$$

Before continuing with the second step of the iteration, the radiance should be measured, that is, an image estimate should be computed from $L^e + \mathcal{T}_1^* L^e$. We can separately calculate the effect of the lightsources on the image and then add the effect of $\mathcal{T}_1^* L^e$. Note that $\mathcal{T}_1^* L^e$ is concentrated in a single point, thus its contribution can be computed by tracing a ray from the eye to this point, and if this point is not occluded, then evaluating the $f_r(\omega_1, \vec{x}, \omega_{eye}) \cdot \Phi$ expression.

The second operator $\mathcal{T}_2^*$ should be applied to

$$L_1 = L^e + \mathcal{T}_1^* L^e,$$

thus both the total power Φ and the probability density have been modified:

$$\Phi_2 = \int_S \int_\Omega L_1(\vec{y}_2, \omega_2) \cos\theta_{\vec{y}_2} \, d\omega_2 \, d\vec{y}_2 = \Phi_1 \cdot (1 + a_{\vec{x}_1}(\omega_1))$$

where $a_{\vec{x}_1}$ is the *albedo* at point $\vec{x}_1$ defined by

$$a_{\vec{x}}(\omega) = \int_\Omega f_r(\omega, \vec{x}, \omega') \cos\theta'_{\vec{x}} \, d\omega',$$

and the new probability density is

$$\frac{d\mathrm{Pr}_2\{\vec{y}_2,\omega_2\}}{d\vec{y}_2 d\omega_2} = \frac{L_1(\vec{y}_2,\omega_2)\cdot\cos\theta_{\vec{y}_2}}{\Phi} =$$

$$\frac{L^e(\vec{y}_2,\omega_2)\cdot\cos\theta_{\vec{y}_2} + f_r(\omega_1,\vec{y}_2,\omega_2)\cos\theta_{\vec{y}_2}\cdot\delta(\vec{y}_2-\vec{x}_1)}{\Phi_1(1+a_{\vec{x}_1}(\omega_1))}.$$

Sampling according to this mixed, discrete-continuous probability density can be realized in the following way. First it is decided randomly whether we sample L^e or the newly generated point using probabilities $1/(1+a_{\vec{x}_1}(\omega_1))$ and $a_{\vec{x}_1}(\omega_1)/(1+a_{\vec{x}_1}(\omega_1))$, respectively. If L^e is selected, then the sampling process is the same as before, i.e. a random point and a random direction are found with probability density

$$\frac{L^e(\vec{y}_2,\omega_2)\cos\theta_{\vec{y}_2}}{\Phi_1}.$$

However, if the new point is chosen, then the direction of the next transfer is found with probability density

$$\frac{f_r(\omega_1,\vec{y}_2,\omega_2)\cos\theta_{\vec{y}_2}}{a_{\vec{x}_1}(\omega_1)}.$$

In either case, a ray defined by the selected point and direction is traced, and the complete power $\Phi_2 = \Phi_1\cdot(1+a_{\vec{x}_1}(\omega_1'))$ is transferred to that point which is hit by the ray. The subsequent steps of the iteration are similar.

Interestingly this iteration is a sequence of variable length random walks, since at each step the point that is last hit by the ray is only selected with a given probability as the starting point of the next ray. The algorithm selects a point from a lightsource and then starts a random walk. The walk finishes after each step with probability $1/(1+a_{\vec{x}_i}(\omega_i))$ and also when the ray hits no object. If a walk finishes, another walk is initiated from the lightsource. When the walk is continued, the transferred power is weighted by $(1+a_{\vec{x}_i}(\omega_i))$, which provides unbiased estimate even if less number of samples are used to simulate higher order bounces. This technique is called the *Russian roulette*[1, 32].

5. Ray-bundle based transfer

In this section another alternative of the random transport operator is introduced, which results in a new and efficient global illumination algorithm. The random approximation of the transport operator transfers the radiance of all surface points of the scene in a single random direction.

In order to store the temporary radiance during the iteration, finite element techniques are used, that tessellate the surfaces into elementary planar patches and assume that a patch has uniform radiance in a given direction (note that this does not mean that the patch has the same radiance in every direction, thus the non-diffuse case can also be handled). According to the concept of finite-elements, the radiance, the emission and the BRDF of patch i are assumed to be independent of the actual point inside the patch, and are denoted by $L_i(\omega)$, $L_i^e(\omega)$ and $\tilde{f}_i(\omega,\omega')$, respectively. It means that the radiance function is approximated in the following form:

$$L(\vec{x},\omega) \approx \sum_i L_i(\omega)\cdot b_i(\vec{x}), \tag{23}$$

where $b_i(\vec{x})$ is 1 on patch i and 0 otherwise. The $L_i(\omega)$ patch radiance can be considered as the average of the radiances of the points on the patch. Since the application of the random transport operator may result in a radiance that is not in the form of equation (23), a "projected" transport operator $\mathcal{T}_F$ should be used that is extended by this averaging operation (formally it is a projection to an adjoint base):

$$(\mathcal{T}_F L)|_i(\omega) = \frac{1}{A_i}\cdot\int_{A_i}\mathcal{T}L(\vec{x},\omega)\,d\vec{x} =$$

$$\frac{1}{A_i}\cdot\int_\Omega\int_{A_i} L(h(\vec{x},-\omega'),\omega')\cdot\cos\theta'\cdot\tilde{f}_i(\omega',\omega)\,d\vec{x}\,d\omega'. \tag{24}$$

Taking into account that the integrand of the inner surface integral is piece-wise constant, it can also be presented in closed form:

$$\int_{A_i} L(h(\vec{x},-\omega'),\omega')\cdot\cos\theta'\cdot\tilde{f}_i(\omega',\omega)\,d\vec{x} =$$

$$\sum_{j=1}^{n}\tilde{f}_i(\omega',\omega)\cdot A(i,j,\omega')\cdot L_j(\omega'), \tag{25}$$

where $A(i,j,\omega')$ expresses the projected area of patch j that is visible from patch i in direction ω'. In the unoccluded case this is the intersection of the projections of patch i and patch j onto a plane perpendicular to ω'. If occlusion occurs, the projected areas of other patches that are in between patch i and patch j should be subtracted as shown in figure 3.

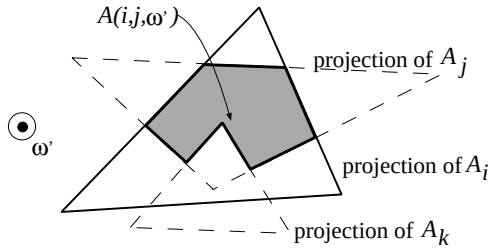

Figure 3: *Interpretation of* $A(i,j,\omega')$

Using equation (25) the projected transport operator can be obtained as:

$$(\mathcal{T}_F L)|_i(\omega) = \int_\Omega\sum_{j=1}^{n}\tilde{f}_i(\omega',\omega)\cdot\frac{A(i,j,\omega')}{A_i}\cdot L_j(\omega')\,d\omega'. \tag{26}$$

Note that equation (26) is highly intuitive as well. The radiance of a patch is the sum of the emission and the reflection of all incoming radiances. The role of the patch-direction-patch "form-factors" is played by $A(i,j,\omega')/A_i$.

Let us define a random operator $\mathcal{T}^*$ that behaves like the projected transport operator in the average case in the following way:

A random direction is selected using a uniform distribution and the radiance of all patches is transferred into this direction.

Formally, the definition is

$$(\mathcal{T}^*(\omega')L(\omega'))|_i(\omega) = 4\pi \cdot \sum_{j=1}^{n} \tilde{f}_i(\omega',\omega) \cdot \frac{A(i,j,\omega')}{A_i} \cdot L_j(\omega'), \tag{27}$$

If the directions ω' is sampled from a uniform distribution, then according to equation (26) the expectation value of the application of this operator is

$$E[(\mathcal{T}^*(\omega')L(\omega'))|_i(\omega)] =$$

$$\int_\Omega (\mathcal{T}^*(\omega')L(\omega'))|_i(\omega)\ \frac{d\omega'}{4\pi} = (\mathcal{T}_F L)|_i(\omega).$$

In the definition of the random operator ω is the actually generated and ω' is the previously generated directions. Thus a "randomization point" is a global direction in this method.

The resulting algorithm is quite simple. In a step of the stochastic iteration an image estimate is computed by reflecting the previously computed radiance estimate towards the eye, and a new direction is found and this direction together with the previous direction are used to evaluate the random transport operator. The complete algorithm — which requires just one variable for each patch *i*, the previous radiance $L[i]$ — is summarized in the following:

```
Clear Image
Generate the first random global direction ω1
for each patch i do L[i] = L_i^e(ω1)
for m = 1 to M do                          // iteration cycles
    Calculate the image estimate relfecting
        L[1], L[2], ... L[n] from ωm towards the eye
    Average the estimate with the Image
    Generate random global direction ωm+1
    for each patch i do
        L^new[i] = L_i^e(ωm+1) +
                4π · Σ_{j=1}^{n} f̃_i(ωm, ωm+1) · A(i,j,ωm)/A_i · L[j]
    endfor
endfor
Display Image
```

There are basically two different methods to calculate the image estimate. On the one hand, evaluating the BRDF once for each patch a radiance value is assigned to them using

$$L^{\text{eye}}[i] = L_i^e(\omega_{\text{eye}}) + 4\pi \cdot \sum_{j=1}^{n} \tilde{f}_i(\omega_m,\omega_{\text{eye}}) \cdot \frac{A(i,j,\omega_m)}{A_i} \cdot L[j],$$

then in order to avoid "blocky" appearance, bi-linear smoothing (i.e. Gouraud shading) can be applied.

Using Phong interpolation, on the other hand, the radiance is evaluated at each point visible through a given pixel using the incoming radiance field, the surface normal and the BRDF of the found point. In order to speed up this procedure, the surface visible at each pixel, the visibility direction and the surface normal can be determined in a preprocessing phase and stored in a map. Phong interpolation is more time consuming but the generated image is not only numerically precise, but is also visually pleasing.

Note that due to the inherent symmetry $A(i,j,\omega') = A(j,i,-\omega')$, the algorithm can easily be generalized to simultaneously handle bi-directional transfers.

6. Calculation of the radiance transport in a single direction

To evaluate the transport operator, we need to know which patches are visible from a given patch, and then we have to weight the radiances of visible patches by the ratio of their visible sizes and the size of the given patch.

This requires the solution of a global visibility problem, where the eye position visits all surface points but the viewing direction is fixed to the selected random direction. This fixed direction is called the *transillumination direction*.

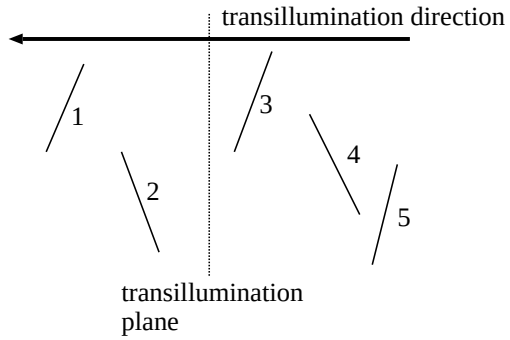

Figure 4: *Global visibility algorithms*

Looking at figure 4, it is easy to see that the global visibility problem can be solved in an incremental way if the patches are visited in the order of their position in the transillumination direction. In fact, what is visible from a patch differs just in a single patch from what is visible from the next patch. This single patch may appear as a new and may hide other patches (figure 5). The required sorting is not obvious if the patches overlap in the transillumination direction,

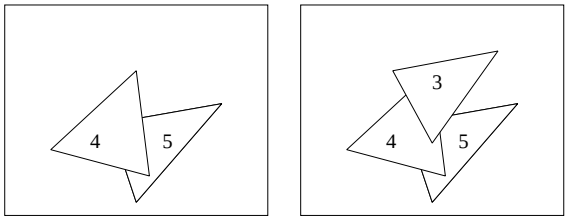

Image seen from patch 3 Image seen from patch 2

Figure 5: *Scene as seen from two subsequent patches*

but this can be solved in a way as proposed in the painter's algorithm[22]. On the other hand, in our case the patches are usually small, thus simply sorting them by their center introduces just a negligible error.

At a given point of all global visibility algorithms the objects visible from the points of a patch must be known. This information is stored in a data structure called the *visibility map*. The visibility map can also be regarded as an image on the plane perpendicular to the transillumination direction. This plane is called the *transillumination plane* (figure 4).

The algorithms that generate the visibility map can be either discrete or continuous. For *continuous algorithms*, the visibility map identifies those regions in which the visibility information is homogeneous (either the same patch is seen or no patch is seen). These algorithms require sophisticated data structures and computational geometry methods not to have cubic time complexity [36].

Discrete algorithms, which decompose the transillumination plane to small pixels of size δP, can solve the problem much faster. For discrete algorithms, the visibility map is simply a rasterized image where each pixel can store either the index of the visible patch or the radiance of the visible point. Discretization may introduce errors. Having examined this phenomenon both theoretically and numerically, we concluded that the stochastic nature of the algorithm can compensate for this error if the projected patch sizes do not fall below the pixel size[36]. Even subpixel patches can be handled using Russian roulette.

Discrete algorithms determine the image of the visible patches through a discretized window assuming the eye to be on patch *i*, the window to be on the transillumination plane and the color of patch *j* to be *j* if the patch is facing to patch *i* and to be 0 otherwise. In [40] we proposed the application of the hardware z-buffer to calculate these images. In this paper, on the other hand, we propose a software solution based on the painter's algorithms, that is more robust and can be faster for complex scenes than the z-buffer based implementation.

Suppose that the patches are sorted in the transillumination direction. If the patches are processed in this order, the

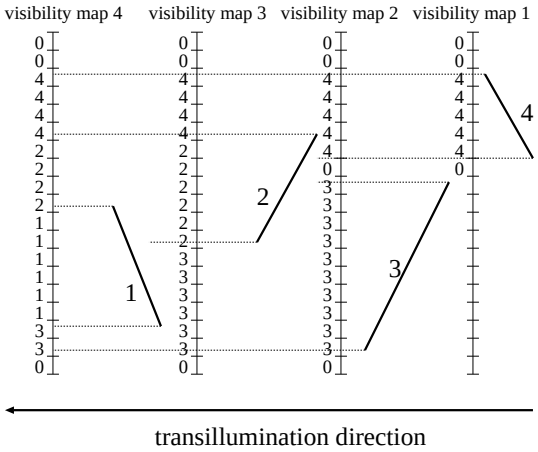

Figure 6: *Application of painter's algorithm*

computation of $A(i, j, \omega')$ requires the determination of the pixel values inside the projection of patch *i*. Then, to proceed with the next patch in the given order, the pixels covered by patch *i* are filled with *i* if patch *i* is not front facing and 0 otherwise. The two steps can be done simultaneously by a modified scan-conversion algorithm that reads the value of the image buffer before modifying it.

This is summarized in the following algorithm [38]:

```
Sort patches in direction ω' (painter's algorithm)
Clear image
for each patch i in the sorted order do
    if patch i is front facing then
      for each pixel of patch i
          j = Read pixel
          A(i, j, ω') += δP
          Write 0 to the pixel
      endfor
    else Render patch i with color i
endfor
```

The average time complexity of this algorithm is $O(N \log N)$ due to the sorting.

The algorithm can easily be extended for sky-light illumination by initializing the image on the transillumination plane by a special value if the direction points downwards (sky is usually above the horizon). When the radiance is transferred and this value is found in a given pixel, then the irradiance of the receiver is updated with the sky-light.

6.1. Can we use quasi-Monte Carlo techniques in iteration?

Stochastic iteration can also be viewed as a single walk which uses a single sequence of usually 4-dimensional ran-

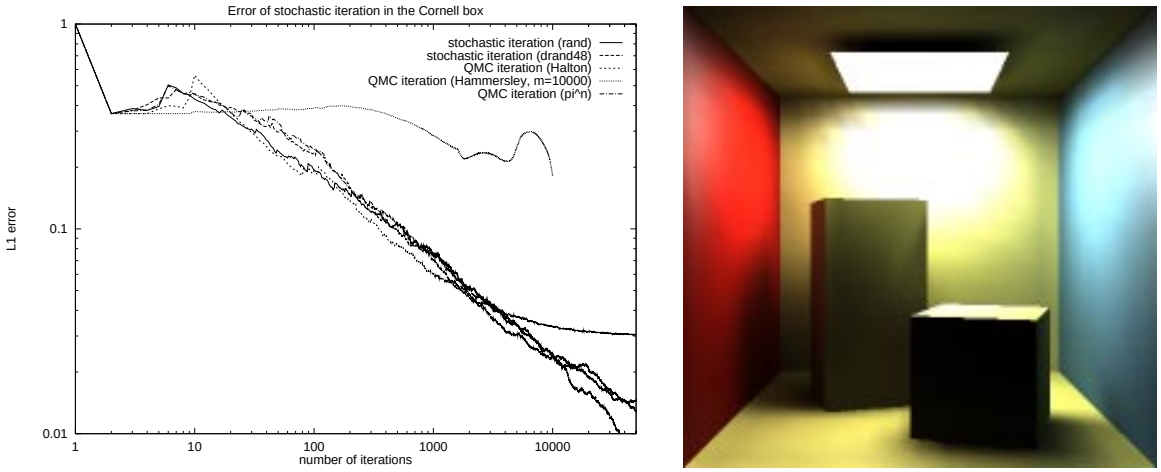

Figure 7: *Ray-bundle based stochastic iteration with random and quasi-random numbers and the test scene*

domization points (for ray-bundle tracing 2-dimensional randomization points), and the $\mathcal{T}^*_{i+k}\mathcal{T}^*_{i+k-1}\dots\mathcal{T}^*_i L^e$ terms are used in integral quadratures simultaneously for all k.

It means that the randomization points should support not only 4-dimensional integration, but using subsequent pairs also 8-dimensional integration, using the subsequent triplets 12-dimensional integration, etc. Sequences that support k-dimensional integrals when subsequent k-tuples are selected are called *k-uniform sequences*[15]. The widely used Halton or Hammersley sequences are only 1-uniform, thus theoretically they should provide false results.

This is obvious for the Hammersley sequence, in which the first coordinate is increasing. Such a sequence would search for only those multiple reflections where the angle corresponding to the first coordinate always increases in subsequent reflections. It is less obvious, but is also true for the Halton sequence. Due to its construction using radical inversion, the subsequent points in the sequence are rather far, thus only those reflections are considered, where the respective angle changes drastically.

In order to avoid false results without getting rid of the quasi-Monte Carlo sequences, [39] proposed the random scrambling of the sample points. A similar problem arises, for example, when generating uniform distributions on a sphere, for which [5] proposed to increase the dimension of the low-discrepancy sequence. Note that this problem is specific to quasi-Monte Carlo integration and does not occur when classical Monte-Carlo method is used to select the sample points (a random sequence is ∞-uniform[15]).

To demonstrate these problems, we tested the ray-bundle based iteration for different random (i.e. pseudo-random) and low-discrepancy sequences. The test scene was the Cornell box. In figure 7 we can see that the Hammersley sequence gives completely wrong result and the Halton sequence also deteriorates from the real solution. The two random generators (rand and drand48), however, performed quite well.

The figure also included a modification of the $q_n = \{\pi^n\}$ quasi-Monte Carlo sequence (operator $\{\}$ selects the fractional part of a number). This is believed to be (but has not been proven to be) ∞-uniform[10]. However, this sequence is very unstable numerically, therefore we used the fractional part of the $q'_n = (\pi - 2) \cdot q'_{n-1} \bmod 100000$ scheme.

7. Simulation results

The presented algorithm has been implemented in C++ in OpenGL environment.

Figure 8 shows a scene as rendered after 500 iterations and after 3000 iterations when the algorithm is fully converged. This pair of images demonstrates that this algorithm can provide good image quality even after relatively few number of iterations. The scene contains specular, metallic objects tessellated to 9519 patches. The calculation of the left image took 9 minutes on a SGI Indigo2 computer.

Figure 9 shows a scene containing 56745 patches after 300 stochastic iterations, which provide an accuracy within 5 percents (30 minutes computation time). The speed of convergence has been measured for the sphere-flake (figure 9) and for the room containing a Beethoven and a teapot (figure 8). The measurement results are shown in figure 10. Note that the algorithm converges faster for sphere-flake scene, which is due to the larger lightsources.

Figure 11 shows a fractal terrain containing 59614 patches with different lighting conditions and wave sizes (45 minutes computation time).

Figure 8: *A scene of a Beethoven and a teapot rendered by stochastic iteration after 500 iterations (left) and when fully converged (right)*

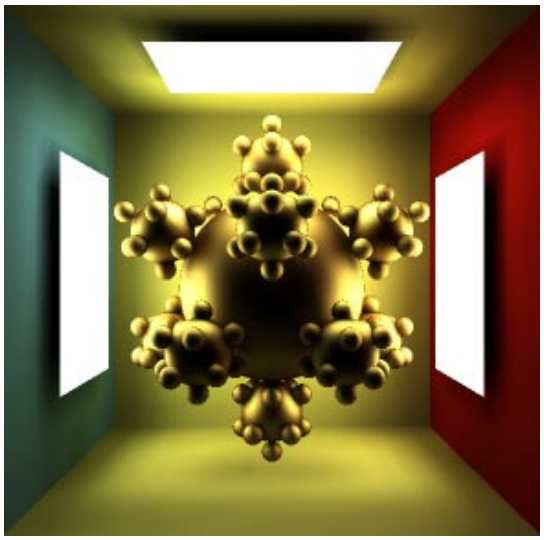

Figure 9: *A golden sphere-flake illuminated by area light-sources*

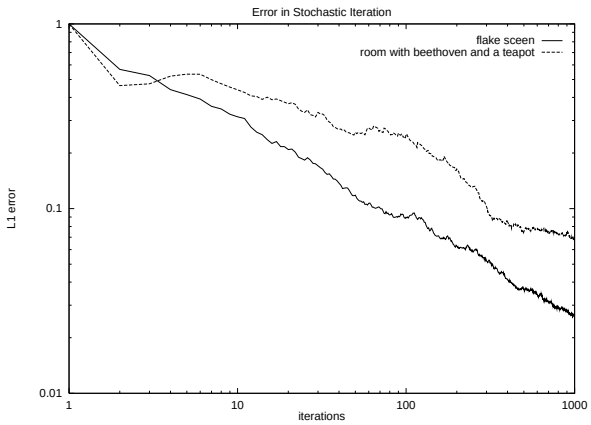

Figure 10: *Convergence of stochastic iteration for the two described scenes*

8. Conclusions

This paper presented a general methodology called stochastic iteration to solve the rendering problem of complex scenes including also glossy surfaces. In a particular realization of this method, we applied ray-bundle tracing that forms bundles of parallel rays that can be traced efficiently using, for example, the painter's algorithm.

The memory requirement of the new method is comparable to that of the diffuse radiosity algorithms although the new algorithm is also capable to handle non-diffuse reflections or refractions.

The time complexity of the algorithm depends on the used global visibility algorithm. Since the global painter's algorithm has $O(n \log n)$ average time complexity (n is the number of patches), the resulting algorithm is superior to the classical, non-hierarchical radiosity algorithms that have $O(n^2)$ complexity. The computation time for a single global radiance transfer depends on the total surface area and the

resolution of the visibility map. Since only the expected value of the visible surface area should be accurately computed, and an actual rasterization can be very coarse, lower resolution visibility maps can also be used. The limit is when the projected patch size becomes comparable to the pixel size, since classical filling algorithms always generate an approximation whose height and width are at least 1. This problem can be solved by modifying the filling algorithm to handle patches or spans randomly if their width or height is less than 1. These low resolution maps increase the variance, thus slow down the convergence a little bit, but still provide unbiased results and significantly reduce the computation time of a single transfer.

Considering the future improvements, we intend to incorporate an adaptive tessellation method into the algorithm. Note that when computing the radiance transport for a given patch, the variation of the incoming radiance can also be easily estimated. If this variation exceeds a given limit, then we cannot assume that the outgoing radiance of the patch is homogeneous, thus the patch has to be subdivided. This tessellation scheme is much more robust than those methods which examine the radiosity gradient. This tessellation scheme also detects highlights that are completely inside a patch and can even provide information where the patch should be subdivided. Considering this, the method is also able to do *discontinuity meshing*.

The algorithm seems to be particularly efficient to render participating media since it can handle very many parallel lines simultaneously. Participating media can be modeled as a set of partly transparent "points" that always project onto a single pixel in the visibility map. Since this point field can be rendered very quickly, the radiance transfer in a single direction can be computed very efficiently.

Since the iteration is basically view-independent, just the result of each iteration step is projected to the eye, images for many cameras can be computed simultaneously, which can be used to produce animations. Since the radiance is stored in the object space, if the surfaces are not highly specular, then the same radiance information remains valid for a wider range of viewing directions. It means that when producing camera animation, the images must be computed by stochastic iteration just for a few camera orientations (viewing directions), and for the inbetweening frames, the radiance of the patches can be interpolated. In order to save space, the radiance is stored just for a few directions at less specular surfaces.

9. Acknowledgements

This work has been supported by the National Scientific Research Fund (OTKA ref.No.: F015884, T029135), the ÖAD (ref.No.: 32öu9) and the Spanish-Hungarian Fund (ref.No.: E9).

References

1. J. Arvo and D. Kirk. Particle transport and image synthesis. In *Computer Graphics (SIGGRAPH '90)*, pages 63–66, 1990.
2. L. Aupperle and P. Hanrahan. A hierarchical illumination algorithms for surfaces with glossy reflection. *Computer Graphics (SIGGRAPH '93)*, pages 155–162, 1993.
3. P. Bekaert, L. Neumann, A. Neumann, M. Sbert, and Y. Willems. Hierarchical Monte-Carlo radiosity. In *Rendering Techniques '98*, pages 259–268, 1998.
4. C. Buckalew and D. Fussell. Illumination networks: Fast realistic rendering with general reflectance functions. *Computer Graphics (SIGGRAPH '89)*, 23(3):89–98, July 1989.
5. F. Castro, R. Martinez, and M. Sbert. Quasi Monte-Carlo and extended first-shot improvements to the multi-path method. In *Spring Conference on Computer Graphics '98*, pages 91–102, 1998.
6. P. H. Christensen, D. Lischinski, E. J. Stollnitz, and D. H. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1):3–33, 1997.
7. P. H. Christensen, E. J. Stollnitz, D. H. Salesin, and T. D. DeRose. Global illumination of glossy environments using wavelets and importance. *ACM Transactions on Graphics*, 15(1):37–71, 1996.
8. Michael Cohen and Donald Greenberg. The hemi-cube, a radiosity solution for complex environments. In *Computer Graphics (SIGGRAPH '85)*, pages 31–40, 1985.
9. R. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. In *Computer Graphics (SIGGRAPH '84)*, pages 137–145, 1984.
10. I. Deák. *Random Number Generators and Simulation.* Akadémia Kiadó, Budapest, 1989.
11. P. Dutre, E. Lafortune, and Y. D. Willems. Monte Carlo light tracing with direct computation of pixel intensities. In *Compugraphics '93*, pages 128–137, 1993.
12. P. Hanrahan, D. Salzman, and L. Aupperle. Rapid hierachical radiosity algorithm. *Computer Graphics (SIGGRAPH '91)*, 1991.
13. D. S. Immel, M. F. Cohen, and D. P. Greenberg. A radiosity method for non-diffuse environments. In *Computer Graphics (SIGGRAPH '86)*, pages 133–142, 1986.
14. J. T. Kajiya. The rendering equation. In *Computer Graphics (SIGGRAPH '86)*, pages 143–150, 1986.
15. D.E. Knuth. *The art of computer programming. Volume 2 (Seminumerical algorithms).* Addison-Wesley, Reading, Mass. USA, 1981.

16. E. Lafortune and Y. D. Willems. Bi-directional path-tracing. In *Compugraphics '93*, pages 145–153, Alvor, 1993.

17. E. Lafortune and Y. D. Willems. A 5D tree to reduce the variance of Monte Carlo ray tracing. In *Rendering Techniques '96*, pages 11–19, 1996.

18. L. Neumann. Monte Carlo radiosity. *Computing*, 55:23–42, 1995.

19. L. Neumann, M. Feda, M. Kopp, and W. Purgathofer. A new stochastic radiosity method for highly complex scenes. In *Proc. of the 5th. EG Workshop on Rendering*, 1994.

20. L. Neumann, A. Neumann, and P. Bekaert. Radiosity with well distributed ray sets. *Computer Graphics Forum (Eurographics'97)*, 16(3):261–270, 1997.

21. L. Neumann, W. Purgathofer, R. F. Tobler, A. Neumann, P. Elias, M. Feda, and X. Pueyo. The stochastic ray method for radiosity. In *Rendering Techniques '95*, pages 206–218, 1995.

22. M. E. Newell, R. G. Newell, and T. L. Sancha. A new approach to the shaded picture problem. In *Proceedings of the ACM National Conference*, pages 443–450, 1972.

23. S. N. Pattanik and S. P. Mudur. Adjoint equations and random walks for illumination computation. *ACM Transactions on Graphics*, 14(1):77–102, 1995.

24. A. Rényi. *Wahrscheinlichkeitsrechnung.* VEB Deutscher Verlag der Wissenschaften, Berlin, 1962.

25. M. Sbert. *The Use of Global Directions to Compute Radiosity*. PhD thesis, Catalan Technical University, Barcelona, 1996.

26. M. Sbert, R. Martinez, and X. Pueyo. Gathering multipath: a new Monte-Carlo algorithm for radiosity. In *Winter School of Computer Graphics '98*, pages 331–338, Plzen, Czech Republic, 1998.

27. M. Sbert, X. Pueyo, L. Neumann, and W. Purgathofer. Global multipath Monte Carlo algorithms for radiosity. *Visual Computer*, pages 47–61, 1996.

28. P. Schröder, S.J. Gortler, M.F. Cohen, and P. Hanrahan. Wavelet projections for radiosity. *Computer Graphics Forum*, 13(2):141–151, 1994.

29. P. Shirley. A ray-tracing method for illumination calculation in diffuse-specular scenes. In *Proc. Graphics Interface*, pages 205–212, 1990.

30. P. Shirley. Time complexity of Monte-Carlo radiosity. In *Eurographics '91*, pages 459–466. Elsevier Science Publishers, 1991.

31. P. Shirley, C. Wang, and K. Zimmerman. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, 1996.

32. F. Sillion and Puech C. *Radiosity and Global Illumination*. Morgan Kaufmann Publishers, Inc., San Francisco, 1994.

33. F. Sillion, G. Drettakis, and C. Soler. Clustering algorithm for radiance calculation in general environments. In *Rendering Techniques '95*, pages 197–205, 1995.

34. I. Sobol. *Die Monte-Carlo Methode.* Deutscher Verlag der Wissenschaften, 1991.

35. M. Stamminger, Slussalek P., and H-P. Seidel. Three point clustering for radiance computations. In *Rendering Techniques '98*, pages 211–222, 1998.

36. L. Szirmay-Kalos. Global ray-bundle tracing. Technical Report TR-186-2-98-18, Institute of Computer Graphics, Vienna University of Technology, 1998. www.cg.tuwien.ac.at/.

37. L. Szirmay-Kalos. Stochastic iteration for non-diffuse global illumination. Technical Report TR-186-2-98-21, Institute of Computer Graphics, Vienna University of Technology, 1998. www.cg.tuwien.ac.at/.

38. L. Szirmay-Kalos and T. Fóris. Sub-quadratic radiosity algorithms. In *Winter School of Computer Graphics '97*, pages 562–571, Plzen, Czech Republic, 1997.

39. L. Szirmay-Kalos, T. Fóris, L. Neumann, and B. Csébfalvi. An analysis to quasi-Monte Carlo integration applied to the transillumination radiosity method. *Computer Graphics Forum*, 16(3):271–281, 1997.

40. L. Szirmay-Kalos and W. Purgathofer. Global ray-bundle tracing with hardware acceleration. In *Rendering Techniques '98*, pages 247–258, 1998.

41. E. Veach and L. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Rendering Techniques '94*, pages 147–162, 1994.

42. E. Veach and L. Guibas. Bidirectional estimators for light transport. In *Computer Graphics (SIGGRAPH '95)*, pages 419–428, 1995.

43. E. Veach and L. Guibas. Metropolis light transport. *Computer Graphics (SIGGRAPH '97)*, pages 65–76, 1997.

44. 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. In *Computer Graphics (SIGGRAPH '87)*, pages 311–324, 1987.

45. K. Zimmerman and P. Shirley. A two-pass solution to the rendering equation with a source visibility preprocess. In *Rendering Techniques '95*, pages 284–295, 1995.

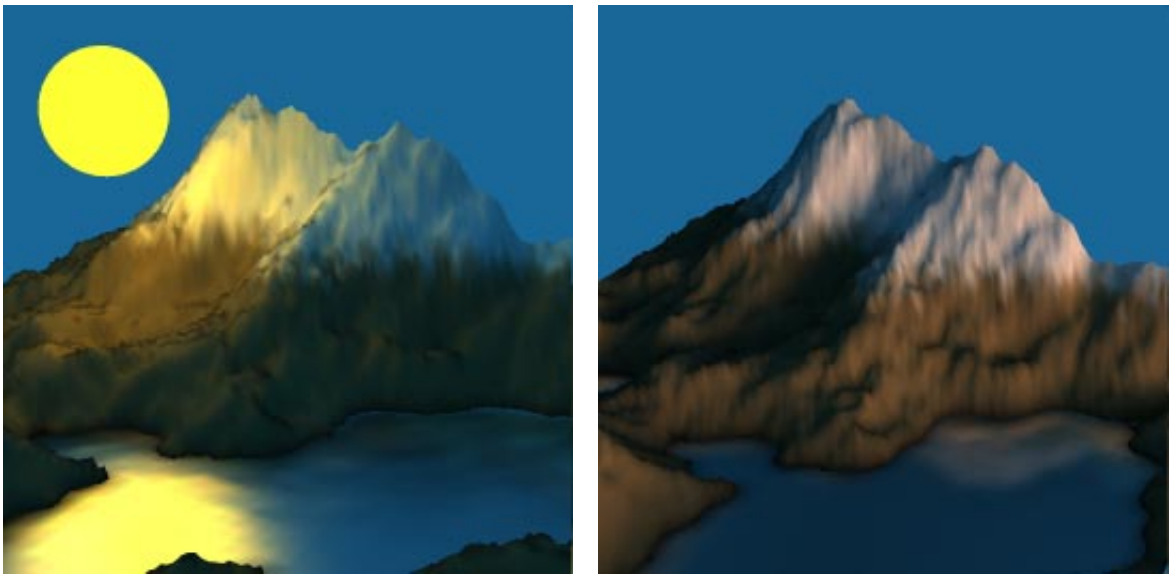

Figure 11: *A mountain with a nearby "moon" and "wavy" water*