Technical Report/TR-186-2-98-23 (*22 August 1998*)

Stochastic Methods in Global Illumination State of the Art Report

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 state of the art report of those global illumination algorithms which involve Monte-Carlo or quasi-Monte Carlo techniques. First it surveys the basic tasks of global illumination, which can be formulated as the solution of either the rendering or the potential equation, then reviews the basic solution techniques, including inversion, expansion and iteration. The paper explains why stochastic approaches are good to solve these integral equations and highlights what kind of fundamental choices we have when designing such an algorithm. It compares, for example, finite-element and continuous methods, pure Monte-Carlo and quasi-Monte Carlo techniques, different versions of importance sampling, Russian roulette, local and global visibility algorithms, etc. Then, a lot of methods are reviewed in a unified framework, that also allows to make comparisons.

Keywords: Rendering equation, potential equation, Monte-Carlo and quasi-Monte Carlo quadratures, finite-element techniques, radiosity, importance sampling, Russian roulette, shooting and gathering random walks, stochastic iteration, Metropolis sampling, distributed ray-tracing, path tracing, photon tracing, light tracing, bi-directional path tracing, photon-map, instant radiosity, global ray-bundle tracing, stochastic ray-radiosity, transillumination method, first-shot, error and complexity

1. Introduction

Generally, the *global illumination problem* is a quadruple[28]

$$\langle S, f_r(\omega', \vec{x}, \omega), L^e(\vec{x}, \omega), \mathbf{W}^e(\vec{x}, \omega)\rangle$$

where S is the geometry of surfaces, f_r is the BRDF of surface points, L^e is the emitted radiance of surface points at different directions and $\mathbf{W}^e$ is a collection of measuring functions.

Global illumination algorithms aim at the modeling and simulation of multiple light-surface interactions to find out the power emitted by the surfaces and landing at the measuring devices after some reflections.

A light-surface interaction can be formulated by the *rendering equation* or alternatively by its adjoint equation, called the *potential equation*.

The *rendering equation*[26] expresses the *radiance* $L(\vec{x}, \omega)$ $[W \cdot m^{-2} \cdot sr^{-1}]$ of a surface point $\vec{x}$ in direction ω, and has the following form:

$$L = L^e + \mathcal{T}L. \tag{1}$$

If only direct contribution is considered, then $L = L^e$. The light-surface interaction is described by integral operator $\mathcal{T}$, which has the following form

$$(\mathcal{T}L)(\vec{x}, \omega) = \int_\Omega L(h(\vec{x}, -\omega'), \omega') \cdot f_r(\omega', \vec{x}, \omega) \cdot \cos\theta' \, d\omega' \tag{2}$$

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 the incoming direction $-\omega'$ (figure 1).

The *potential equation*[42], on the other hand, uses the *potential* $W(\vec{y}, \omega')$ as a fundamental measure, which expresses the effect of emitting unit power from $\vec{y}$ in direction ω' on

© The Institute of Computer Graphics, Vienna University of Technology.

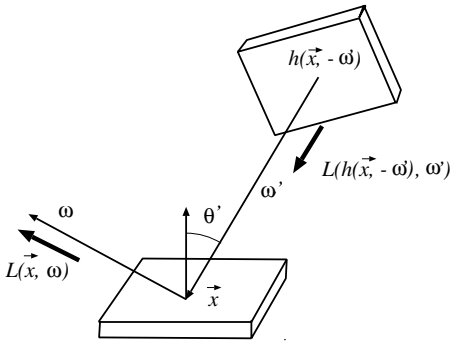

Figure 1: *Geometry of the rendering equation*

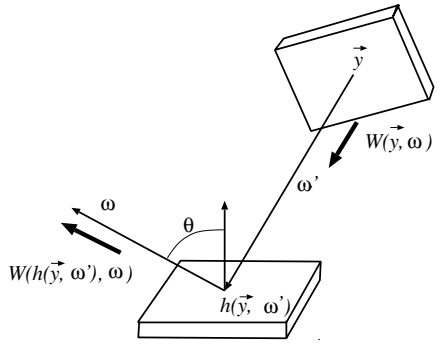

Figure 2: *Geometry of the potential equation*

a measuring device having sensitivity $W^e(\vec{y},\omega')$ (for example, this device can measure the power going through a single pixel of the image, or leaving a surface element at any direction). If only direct contribution is considered, then $W(\vec{y},\omega') = W^e(\vec{y},\omega')$. To take into account light reflections, we can establish the potential equation

$$W = W^e + \mathcal{T}'W. \tag{3}$$

In this equation integral operator $\mathcal{T}'$ — which is the adjoint of $\mathcal{T}$ — describes the potential transport

$$(\mathcal{T}'W)(\vec{y},\omega') =$$

$$\int_\Omega W(h(\vec{y},\omega'),\omega)\cdot f_r(\omega', h(\vec{y},\omega'),\omega)\cdot\cos\theta\; d\omega, \tag{4}$$

where θ is the angle between the surface normal and the outgoing direction ω.

According to the definition of the radiance

$$L(\vec{y},\omega) = \frac{d\Phi(\vec{y},\omega)}{d\vec{y}\, d\omega\cos\theta},$$

the power detected by a measuring device can be computed by the measuring function from the radiance

$$\int_S\int_\Omega d\Phi(\vec{y},\omega)\cdot W^e(\vec{y},\omega) =$$

$$\int_S\int_\Omega L(\vec{y},\omega)\cos\theta\cdot W^e(\vec{y},\omega)\; d\vec{y}\, d\omega = \mathcal{M}L, \tag{5}$$

where $\mathcal{M}$ is the radiance measurement operator. Having introduced the scalar product $\langle u, v\rangle$

$$\langle u, v\rangle = \int_S\int_\Omega u(\vec{y},\omega)\cdot v(\vec{y},\omega)\; d\vec{y}\, d\omega,$$

and the cosine weighted scalar product $\langle u, v\rangle_{\cos}$

$$\langle u, v\rangle_{\cos} = \langle u\cdot\cos\theta, v\rangle = \langle u, v\cdot\cos\theta\rangle,$$

we can obtain an alternative form of the measurement operator

$$\mathcal{M}L = \langle L, W^e\rangle_{\cos}.$$

A simple measurement function for a pinhole camera is

$$W^e(\vec{y},\omega) = \frac{\delta(\omega-\omega_f)}{\cos\theta}\cdot\xi(h(\vec{y},\omega))$$

where ω_f is the focal point and $\cos\theta$ is the cosine angle between the normal of the visible surface and the viewing direction. With this measurement function, the power going through a pixel of area P can be obtained using equation (5):

$$\int_{S_P} L(h(\vec{p},-\omega_{\vec{p}}),\omega_{\vec{p}})\cdot\xi(\vec{p})\; d\vec{p}, \tag{6}$$

where S_P is the support of ξ. S_P is usually, but not necessarily, equal to the pixel surface.

Alternatively to the radiance, the power arriving at the measuring device can also be computed from the potential:

$$\int_S\int_\Omega d\Phi^e(\vec{y},\omega')\cdot W(\vec{y},\omega') =$$

$$\int_S\int_\Omega W(\vec{y},\omega')\cdot L^e(\vec{y},\omega')\cdot\cos\theta\; d\vec{y}\, d\omega' = \mathcal{M}'W, \tag{7}$$

where $\mathcal{M}'$ is the potential measuring operator. Note that unlike the radiance measuring operator, the potential measuring operator integrates on the lightsource.

This measuring operator can also be given in a scalar product form

$$\mathcal{M}'W = \langle L^e, W\rangle_{\cos}. \tag{8}$$

Since the rendering or the potential equation contain the unknown radiance function both inside and outside the integral, in order to express the solution, this coupling should

© Institute of Computer Graphics 1998

be resolved. The possible solution techniques fall into one of the following three categories: *inversion*, *expansion* and *iteration*.

Operator $\mathcal{T}$ represents light-surface interaction, thus each of its application generates a higher-bounce estimate of the light transport (or alternatively $\mathcal{T}'$ represents potential-surface interaction). For physically plausible optical material models, a reflection or refraction always decreases the total energy, thus the integral operator is always a contraction. However, when the transport is evaluated numerically, computation errors may pose instability problems if the scene is highly reflective. As we shall see, expansion and iteration exploit the contractive property of the transport operator, but inversion does not.

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:

$$(1-\mathcal{T})L = L^e \implies L = (1-\mathcal{T})^{-1}L^e. \quad (9)$$

Thus the measured power is

$$\mathcal{M}L = \mathcal{M}(1-\mathcal{T})^{-1}L^e. \quad (10)$$

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. This kind of approximation is provided by finite-element techniques that project the problem into a finite dimensional function space, and approximate the solution here. This projection converts the original integral equation into a system of linear equations, which can be inverted, for example, by Gaussian elimination method. This approach was used in early radiosity methods, but have been ruled out due to the cubic time complexity and the numerical instability of the Gaussian elimination.

Inversion has a unique property that is missing in the other two methods. Its efficiency does not depend on the contractivity of the integral operator, neither does it even require the integral operator to be a contraction.

Since no stochastic alternative has been proposed yet for the deterministic inversion, we do not consider this option any further in this paper.

1.2. Expansion

Expansion techniques eliminate the coupling by obtaining the solution in the form of an infinite Neumann series.

1.2.1. Expansion of the rendering equation: gathering walks

Substituting the right side's L by $L^e + \mathcal{T}L$, which is obviously L according to the equation, we get:

$$L = L^e + \mathcal{T}L = L^e + \mathcal{T}(L^e + \mathcal{T}L) = L^e + \mathcal{T}L^e + \mathcal{T}^2 L. \quad (11)$$

Repeating this step n times, the original equation can be expanded into a Neumann series:

$$L = \sum_{i=0}^{n} \mathcal{T}^i L^e + \mathcal{T}^{n+1} L. \quad (12)$$

If integral operator $\mathcal{T}$ is a contraction, then $\lim_{n\to\infty} \mathcal{T}^{n+1}L = 0$, thus

$$L = \sum_{i=0}^{\infty} \mathcal{T}^i L^e. \quad (13)$$

The measured power is

$$\mathcal{M}L = \sum_{i=0}^{\infty} \mathcal{M}\mathcal{T}^i L^e. \quad (14)$$

The terms of this infinite Neumann series have intuitive meaning as well: $\mathcal{M}\mathcal{T}^0 L^e = L^e$ comes from the emission, $\mathcal{M}\mathcal{T}^1 L^e$ comes from a single reflection, $\mathcal{M}\mathcal{T}^2 L^e$ from two reflections, etc.

In order to understand how this can be used to determine the power going through a single pixel, let us examine the structure of $\mathcal{M}\mathcal{T}^i L^e$ as a single multi-dimensional integral for the $i = 2$ case:

$$\mathcal{M}(\mathcal{T}^2 L^e) = \int_{S_P}\int_{\Omega'_1}\int_{\Omega'_2} w_0(\vec{p}) \cdot w_1(\vec{x}_1) \cdot w_2(\vec{x}_2) \cdot L^e(\vec{x}_3, \omega'_2)\, d\omega'_2 d\omega'_1 d\vec{p}. \quad (15)$$

where

$$\begin{aligned} \vec{x}_1 &= h(\vec{p}, -\omega_{\vec{p}}), \\ \vec{x}_2 &= h(\vec{x}_1, -\omega'_1), \\ \vec{x}_3 &= h(\vec{x}_2, -\omega'_2) = h(h(\vec{x}_1, -\omega'_1), -\omega'_2), \end{aligned} \quad (16)$$

and the weights are

$$\begin{aligned} w_0 &= \xi(\vec{p}), \\ w_1 &= f_r(\omega'_1, \vec{x}_1, \omega_{\vec{p}}) \cdot \cos\theta'_1, \\ w_2 &= f_r(\omega'_2, \vec{x}_2, \omega'_1) \cdot \cos\theta'_2. \end{aligned} \quad (17)$$

Thus to evaluate the integrand at point $(\vec{p}, \omega'_1, \omega'_2)$, the following algorithm must be executed:

1. Point $\vec{x}_1 = h(\vec{p}, -\omega_{\vec{p}})$ that is visible through the point $\vec{p}$ of the pixel from the eye should be found. This can be done by sending a ray from the eye into the direction of $\vec{p}$ and identifying the surface that is first intersected.

© Institute of Computer Graphics 1998

2. Surface point $\vec{x}_2 = h(\vec{x}_1, -\omega_1')$ — that is the point which is visible from $\vec{x}_1$ at direction $-\omega_1'$ — must be determined. This can be done by sending another ray from $\vec{x}_1$ into direction $-\omega_1'$ and identifying the surface that is first intersected.
3. Surface point $\vec{x}_3 = h(h(\vec{x}_1, -\omega_1'), -\omega_2')$ — that is the point visible from $\vec{x}_2$ at direction $-\omega_2'$ — is identified. This means the continuation of the ray tracing at direction $-\omega_2'$.
4. The emission intensity of the surface at $\vec{x}_3$ in the direction of ω_2' is obtained and is multiplied with the cosine terms and the BRDFs of the two reflections.

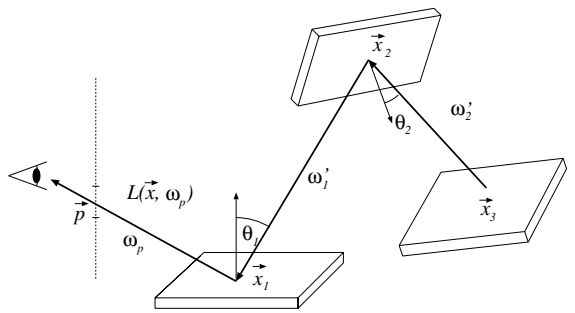

Figure 3: *The integrand of $\mathcal{M}\mathcal{T}^2 L^e$ is a two-step gathering walk*

This algorithm can easily be generalized for arbitrary number of reflections. A ray is emanated recursively from the visible point at direction ω_1' then from the found surface at ω_2', etc. until ω_n'. The emission intensity at the end of the walk is read and multiplied by the BRDFs and the cosine terms of the stages of the walk.

These walks provide the value of the integrand at "point" $\vec{p}, \omega_1', \omega_2', \ldots, \omega_n'$.

Note that a single walk of length n can be used to estimate the 1-bounce, 2-bounce, etc. n-bounce transfer simultaneously, if the emission is transferred not only from the last visited point but from all visited points.

The presented walking technique starts at the eye and *gathers* the illumination encountered during the walk. The gathered illumination is attenuated according to the cosine weighted BRDFs of the path.

So far, we have examined the structure of the terms of the Neumann series as a single multi-dimensional integral. Alternatively, it can also be considered as recursively evaluating many directional integrals. Examining $\mathcal{M}\mathcal{T}^2 L^e$ again:

$$\int_{S_P} w_0 \cdot \left[\int_{\Omega_1'} w_1 \cdot \left[\int_{\Omega_2'} w_2 \cdot L^e \, d\omega_2' \right] d\omega_1' \right] d\vec{p}. \qquad (18)$$

In order to estimate the outer integral of variable $\vec{p}$, the integrand is needed in the sample points $\vec{p}$. This, in turn, requires the estimation of the integral of variable ω_1' at $\vec{p}$, which recursively needs again the approximation of the integral of variable ω_2' at $(\vec{p}, \omega_1')$.

If the same number — say m — of sample points are used for each integral quadrature, then this recursive approach will use m points for the 1-bounce transfer, m^2 for the two-bounces, m^3 for the three-bounces, etc. This kind of subdivision of paths is called *splitting*[2]. Splitting becomes prohibitive for high-order reflections and is not even worth doing because of the contractive property of the integral operator. Due to the contraction, the contribution of higher-order bounces is less thus it is not very wise to compute them as precisely as low-order bounces.

1.2.2. Expansion of the potential equation: shooting walks

The potential equation can also be expanded into a Neumann series similarly to the rendering equation.

$$W = \sum_{i=0}^{\infty} \mathcal{T}'^i W^e, \qquad (19)$$

which results in the following measured power

$$\mathcal{M}'W = \sum_{i=0}^{\infty} \mathcal{M}'\mathcal{T}'^i W^e. \qquad (20)$$

$\mathcal{M}'W^e$ is the power measured by the device from direct emission. $\mathcal{M}'\mathcal{T}'W^e$ is the power after a single reflection, $\mathcal{M}'\mathcal{T}'^2 W^e$ is after two reflections, etc.

Let us again consider the structure of $\mathcal{M}'\mathcal{T}'^2 W^e$:

$$\mathcal{M}'\mathcal{T}'^2 W^e =$$

$$\int_S \int_{\Omega_1} \int_{\Omega_2} \int_{\Omega_3} L^e(\vec{y}_1, \omega_1) \cdot w_0 \cdot w_1 \cdot w_2 \cdot W^e(\vec{y}_2, \omega_{\vec{p}'}) \, d\omega_3 d\omega_2 d\omega_1 d\vec{y}_1 =$$

$$\int_S \int_{\Omega} \int_{\Omega_1} L^e(\vec{y}_1, \omega_1) \cdot w_0(\vec{y}_1) \cdot w_1(\vec{y}_2) \cdot w_2(\vec{y}_3) \, d\omega_2 d\omega_1 d\vec{y}_1. \qquad (21)$$

if $\vec{y}_3$ is visible though the given pixel and 0 otherwise, where

$$\begin{aligned} \vec{y}_2 &= h(\vec{y}_1, \omega_1), \\ \vec{y}_3 &= h(\vec{y}_2, \omega_2) = h(h(\vec{y}_1, \omega_1), \omega_2) \end{aligned} \qquad (22)$$

and the weights are

$$\begin{aligned} w_0 &= \cos\theta_1, \\ w_1 &= f_r(\omega_1, \vec{y}_2, \omega_2) \cdot \cos\theta_2, \\ w_2 &= f_r(\omega_2, \vec{y}_3, \omega_{\vec{p}}) \cdot \xi(\vec{p}). \end{aligned} \qquad (23)$$

Thus to evaluate the integrand at point $(\vec{y}_1, \omega_1, \omega_2)$, the following algorithm must be executed:

© Institute of Computer Graphics 1998

1. The cosine weighted emission of point $\vec{y}_1$ in direction ω_1 is computed. Surface point $\vec{y}_2 = h(\vec{y}_1, \omega_1)$ — that is the point which is visible from $\vec{y}_1$ at direction ω_1 — must be determined. This can be done by sending a ray from $\vec{y}_1$ into direction ω_1 and identifying the surface that is first intersected. This point "receives" the computed cosine weighted emission.
2. Surface point $\vec{y}_3 = h(h(\vec{y}_1, \omega_1), \omega_2)$ — that is the point visible from $\vec{y}_2$ at direction ω_2 — is identified. This means the continuation of the ray tracing at direction ω_2. The emission is weighted again by the local BRDF and the cosine of the normal and the outgoing direction.
3. It is determined whether or not this point $\vec{y}_3$ is visible from the eye, and through which pixel. Then the transferred emission is weighted again by only the local BRDF and the contribution to the pixel is incremented by the weighted emission.

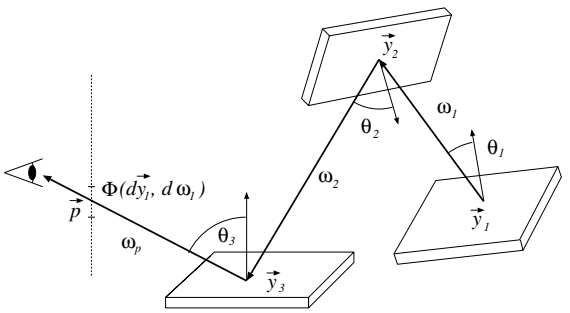

Figure 4: *The integrand of $\mathcal{T}'^2 W^e$ is a two-step shooting walk*

This type of walk, called *shooting*, starts at a known point $\vec{y}_1$ of a lightsource and simulates the photon reflection for a few times and finally arrives at a pixel whose radiance this walk contributes to.

Note that in gathering walks the BRDF is multiplied with the cosine of the angle between the normal and the incoming direction, while in shooting walks with the cosine of the angle between the normal and the outgoing direction. On the other hand, in gathering walks, the cosine angle of the emitting surface is not used, while in shooting walks the cosine angle of the last visible surface is neglected.

1.2.3. Merits and disadvantages of expansion methods

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. Practical implementations usually truncate the infinite Neumann series, which introduces some bias, or stop the walks randomly, which significantly reduces the samples of higher order interreflections. These can result in visible artifacts for highly reflective scenes.

On the other hand, expansion methods also have an important advantage. Namely, they do not require 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.

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

1.3. Iteration

Iteration techniques realize that the solution of integral equation (1) is the fixed point of the following iteration scheme

$$L_n = L^e + \mathcal{T} L_{n-1}, \tag{24}$$

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

$$\mathcal{M}L = \lim_{n \to \infty} \mathcal{M}L_n, \tag{25}$$

In order to store the approximating functions L_n, usually finite-element techniques are applied, as for example, in *diffuse radiosity*[57], or in non-diffuse radiosity using *partitioned hemisphere*[21], *directional distributions*[59] or *illumination networks*[8].

There are two critical problems here. On the one hand, since the domain of L_n 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*[19, 4], *wavelet* or *multiresolution methods*[11, 51] and *clustering*[58, 10, 61] can help, the memory requirements are still prohibitive for complex scenes. This problem is less painful for the diffuse case since here the domain of the radiance is only 2 dimensional.

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[66]. 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.

In the diffuse radiosity setting several methods have been proposed to improve the quality of the iteration. For example, we can use Chebyshev iteration instead of the Jacobi or the Gauss-Seidel method for such ill conditioned systems[5].

© Institute of Computer Graphics 1998

On the other hand, realizing that the crucial part of designing such an the algorithm is finding a good and "small" approximation of the transport operator, the method called *well-distributed ray-sets*[39, 6] proposes the adaptive approximation of the transport operator. This approximation is a set of rays selected carefully taking into account the important patches and directions. In [6], the adaptation of the discrete transport operator is extended to include patch subdivision as well, to incorporate the concepts of *hierarchical radiosity*[19]. The adaptation strategy is to refine the discrete approximation (by adding more rays to the set), when the iteration with the coarse approximation is already stabilized. Since the discrete approximation of the transport operator is not constant but gets finer in subsequent phases, the error accumulation problem can be controlled but is not eliminated.

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

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. Summarizing, iteration is the better option if the scene is not highly specular.

2. Why should we use stochastic methods?

Expansion techniques require the evaluation of very high-dimensional — in fact, infinite dimensional — integrals. When using classical quadrature rules for multi-dimensional integrals [44], such as for example the trapezoidal rule, in order to provide a result with a given accuracy, the number of sample points is in the order of $O(M^D)$, where D is the dimension of the domain. This phenomenon is called the *dimensional core* or *dimensional explosion* and makes classical quadrature rules prohibitively expensive for higher dimensions. The reason of the dimensional explosion is that these rules are usually based on uniform grids — that are simple Cartesian products of the 1D grid in higher dimensions — in which different dimensions do not effectively interact.

However, Monte-Carlo or quasi-Monte Carlo techniques distribute the sample points simultaneously in all dimensions, thus they can avoid dimensional explosion. For example, the probabilistic error bound of Monte-Carlo integration is $O(M^{-0.5})$, independently of the dimension of the domain. D-dimensional low discrepancy series[41] can even achieve $O(\log^D M/M) = O(M^{-(1-\epsilon)})$ convergence rates for finite variation integrands.

Furthermore, classical quadrature cannot be used for infinite dimensional integrals, thus the Neumann series should be truncated after D terms. This truncation introduces a bias of order $\lambda^{D+1} \cdot ||L_e||/(1-\lambda)$. Using a Russian roulette based technique, on the other hand, Monte-Carlo methods are appropriate for even infinite dimensional integrals.

Thus we can conclude that the stochastic approach is indispensable for expansion methods.

The application of randomized techniques in iteration is not so evident, but can also be justified. On the simplest level, these methods also use integration in each iteration step. The dimension of the domain is usually not very high. For example, iterative diffuse radiosity methods need to evaluate 4-dimensional integrals to obtain form factors. The dimension is often reduced to 2 by a brutal simplification, which computes one of the two surface integrals from a single value. For even 4-dimensional integrals Monte-Carlo methods are superior than classical quadratures thus in accurate algorithms they are highly recommended.

Furthermore, when stochastic iteration is applied, the operator should be like the real operator just in the average case. This allows us to use significantly simpler realizations. For example, the integral part of the operator can also be approximated as an expectation value, thus in a single transfer usually no explicit integral is computed. As we shall see, it is relatively easy to apply random operators whose expected case behavior gives exactly back that of the real operator. Thus the error accumulation problem can also be avoided.

If the operator is highly simplified, it does not require the integrand everywhere in the domain, thus a lot of storage space can be saved. Compared to the astronomical storage requirements of non-diffuse radiosity methods, for example, with stochastic iteration we can achieve the same goal with one variable per patch[69]. This argument loses some of its importance when view-independent solution is also required, since the final solution should be stored anyway. This is not a problem if only the diffuse case is considered, since using a single radiosity value per patch the image can be generated from any viewpoint. For the non-diffuse case, the reduced storage gets particularly useful when the image is to be calculated in only a single, or in a few eye positions.

Summarizing, the advantages of stochastic iteration are the simplicity speed, affordable storage requirements and numerical stability even for very large systems containing highly reflective materials.

© Institute of Computer Graphics 1998

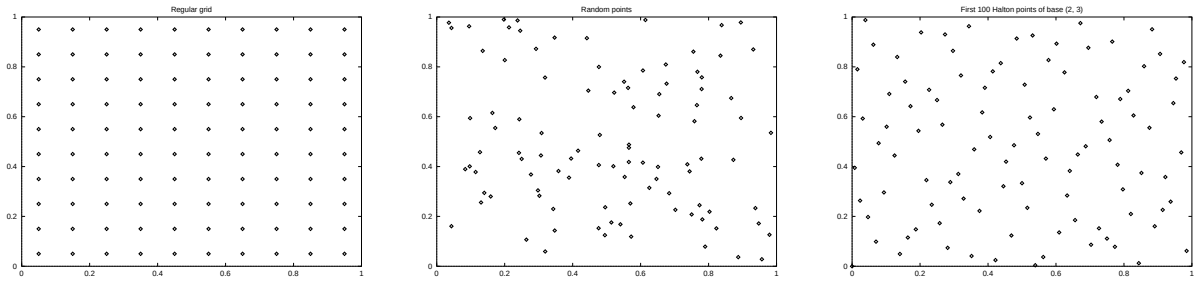

Figure 5: *100 points distributed by a regular grid (left), random distribution (middle) and Halton low-discrepancy sequence (right)*

3. Options in constructing stochastic rendering methods

3.1. Monte-Carlo versus quasi-Monte Carlo

The core of the computations of all methods is the evaluation of high-dimensional integrals (for inversion and iteration it means 4 dimensional integrals, for expansion, it means, at least theoretically, infinite-dimensional integrals). To evaluate an integral, we can use quadrature formulae, that have the following form in the simplest case:

$$\int\limits_{[0,1]^D} f(\mathbf{z})\ d\mathbf{z} \approx \frac{1}{M} \cdot \sum_{i=1}^{M} f(\mathbf{z}_i). \tag{26}$$

Those sets of sample points that provide an exact integral value in the asymptotic sense are called *uniform sequences*.

Well known examples for uniform sequences are the uniform grid or the uniformly distributed random samples (figure 5). The application of random samples can be justified by assuming that the integrand is multiplied by a constant $p(\mathbf{z}) = 1$ function which is the probability density of a uniformly distributed random variable, then realizing that the integral is the formula of the expectation of $f(\mathbf{z})$. Expectations can be approximated by averages if sample points are selected according to probability density $p(\mathbf{z}) = 1$:

$$\int\limits_{[0,1]^D} f(\mathbf{z})\ d\mathbf{z} = \int\limits_{[0,1]^D} f(\mathbf{z}) \cdot p(\mathbf{z})\ d\mathbf{z} = E[f(\mathbf{z})] \approx \frac{1}{M} \cdot \sum_{i=1}^{M} f(\mathbf{z}_i). \tag{27}$$

To find out which are those sample sets that can effectively be used in numerical integration, the *Koksma-Hlawka inequality*[41] gives us some hints (unfortunately, it is valid only for finite-variation functions, but the basic observations are still useful in more general circumstances):

$$\left| \int\limits_{\mathbf{z} \in [0,1]^D} f(\mathbf{z})\ d\mathbf{z} - \frac{1}{M} \sum_{i=1}^{M} f(\mathbf{z}_i) \right| \leq \mathcal{V}_{\mathrm{HK}} \cdot \mathcal{D}^*(\mathbf{z}_1, \ldots \mathbf{z}_N), \tag{28}$$

where $\mathcal{V}_{\mathrm{HK}}$ is the *variation* of f in the sense of Hardy and Krause, and $\mathcal{D}^*(\mathbf{z}_1, \ldots \mathbf{z}_N)$ is the *star-discrepancy* of the used sample set (for the bounds and computation of the discrepancy refer to [41, 53, 16]).

According to this inequality, the error can be upper-bounded by the product of two independent factors, the variation of the integrand and the discrepancy of the used samples set. The discrepancy shows how uniformly the set is distributed[53]. This immediately presents two orthogonal strategies to improve the quality of quadratures. Either we try to make the function flat by appropriate variable transformations, or use very uniformly distributed sample sets. The first technique is called *importance sampling*[60], while the second involves the *stratification*[60, 36, 1] of random points or the application of *low-discrepancy series*[41, 82, 44, 29, 60].

Low-discrepancy samples are deterministic point sets that are designed to be optimally uniform, thus replacing the random points by them improves the accuracy of the integral quadrature. Quadrature rules that use low-discrepancy series instead of random points are called *quasi-Monte Carlo methods*.

Quasi-Monte Carlo techniques have been first applied to solve the diffuse rendering equation by Keller[27], where the integrand was generally discontinuous and therefore of infinite variation, thus the superiority of quasi-Monte Carlo method could not been theoretically justified (note that the Koksma-Hlawka inequality is meaningless if the variation is infinite). However, the numerical evidence showed that quasi-Monte Carlo methods can slightly be better than Monte-Carlo techniques.

The quasi-Monte Carlo integration of infinite variation functions has been analyzed in [44, 70], where it was concluded that quasi-Monte Carlo methods are still better but lose their

© Institute of Computer Graphics 1998

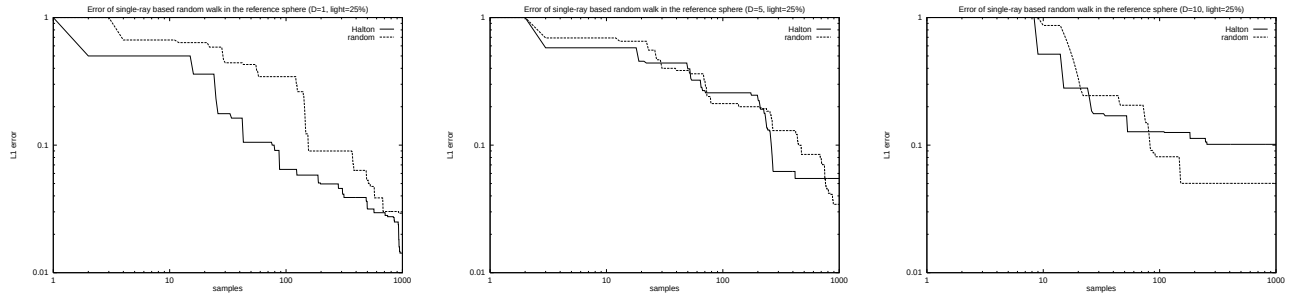

Figure 6: *Error measurements for 1, 5 and 10 bounces*

advantage in higher dimensions. The other important problem is that although a low-discrepancy series has almost linearly decreasing discrepancy in the asymptotic sense, this discrepancy can still be high for not very many points (in the solution of the rendering equation we rarely use more than 1000 samples for the estimation of a single pixel). In the case of the Halton series, for example, the *base* of the series strongly affects the initial behavior of the discrepancy. These base numbers are different prime numbers for different dimensions, thus for high-dimensional integrals the base numbers can be quite high, which results in degraded performance.

To demonstrate this, in figure 6 the errors of different bounces generated by quasi-Monte Carlo and the Monte-Carlo quadratures have been compared for a spherical diffuse scene where only a part is lightsource. For this scene the analytical solution of the rendering equation is possible[20, 70].

3.2. Continuous versus finite-element based methods

Iteration requires the representation of the temporary radiance function L_n. So does expansion if view-independent solution is needed since the final radiance distribution must be represented in a continuous domain.

To represent a function over a continuous domain, finite element methods can be used which approximate the function in the following form:

$$L(\vec{x},\omega) \approx \sum_{j=1}^{n} \mathbf{L}_j \cdot b_j(\vec{x},\omega) = \mathbf{b}^T(\vec{x},\omega) \cdot \mathbf{L} \qquad (29)$$

where $b_j(\vec{x},\omega)$ is a system of predefined basis functions, and $\mathbf{L}_j$ factors are unknown coefficients.

This representation can also be seen as projecting the infinite dimensional space of the possible radiance functions into a finite-dimensional function space defined by the basis functions.

Substituting this approximation into the rendering equation we can obtain:

$$\mathbf{b}^T \cdot \mathbf{L} \approx \mathbf{b}^T \cdot \mathbf{L}^e + \mathcal{T}(\mathbf{b}^T \cdot \mathbf{L}). \qquad (30)$$

Note that equality cannot be guaranteed, since even if $\mathbf{b}^T(\vec{x},\omega) \cdot \mathbf{L}$ is in the subspace defined by the basis functions, the integral operator $\mathcal{T}$ may result in a function that is out of this space. This can be solved by projecting the result back to the subspace and using a projected integral operator $\mathcal{T}_F$ in the following way:

$$\mathcal{T}_F\mathbf{L} = \langle \mathcal{T}\mathbf{b}^T \cdot \mathbf{L}, \tilde{\mathbf{b}} \rangle. \qquad (31)$$

where $\langle \mathcal{T}L, \tilde{\mathbf{b}} \rangle$ is a vector of scalar products

$$\langle \mathcal{T}L, \tilde{b}_1 \rangle, \ldots \langle \mathcal{T}L, \tilde{b}_n \rangle$$

and $\tilde{b}_i$ is an adjoint basis of b_i, since we require that $\langle \tilde{b}_i, b_j \rangle = 1$ if $i = j$ and 0 otherwise.

Since $\mathbf{L}$ is constant, we can also obtain

$$\mathcal{T}_F\mathbf{L} = \langle \mathcal{T}\mathbf{b}^T, \tilde{\mathbf{b}} \rangle \cdot \mathbf{L} = \mathbf{F} \cdot \mathbf{L}, \qquad (32)$$

where $\mathbf{F} = \langle \mathcal{T}\mathbf{b}^T, \tilde{\mathbf{b}} \rangle$ is a matrix, where the i,j element is $\langle \mathcal{T}b_j, \tilde{b}_i \rangle$.

Thus the projection converts the original integral to the following form:

$$\mathbf{L} = \mathbf{L}^e + \mathcal{T}_F\mathbf{L} = \mathbf{L}^e + \mathbf{F} \cdot \mathbf{L}. \qquad (33)$$

An adjoint of this linear equation can be derived by supposing that each basis function b_i is associated with a measurement device W_i^e that measures the power $\mathbf{P}_i$ leaving the support of the basis function. Thus we obtain

$$\langle W_i^e, \mathbf{b}^T \cdot \mathbf{L} \rangle_{\cos} = \langle W_i^e, b_i \rangle_{\cos} \cdot \mathbf{L}_i = \mathbf{P}_i.$$

Similarly, the measured emission power is

$$\langle W_i^e, \mathbf{b}^T \cdot \mathbf{L}^e \rangle_{\cos} = \langle W_i^e, b_i \rangle_{\cos} \cdot \mathbf{L}_i^e = \mathbf{P}_i^e.$$

Applying measurement operator W_i^e for equation (33),

© Institute of Computer Graphics 1998

we can obtain the following equation:

$$\langle W_i^e, b_i\rangle_{\cos}\cdot\mathbf{L}_i = \langle W_i^e, b_i\rangle_{\cos}\cdot\mathbf{L}_i^e + \langle W_i^e, b_i\rangle_{\cos}\cdot\sum_{j=1}^{n}\mathbf{F}_{ij}\mathbf{L}_j \tag{34}$$

This can also be presented in matrix form

$$\mathbf{P} = \mathbf{P}^e + \mathbf{H}\cdot\mathbf{P}, \tag{35}$$

where

$$\mathbf{H}_{ij} = \mathbf{F}_{ij}\cdot\frac{\langle W_i^e, b_i\rangle_{\cos}}{\langle W_j^e, b_j\rangle_{\cos}}. \tag{36}$$

When finite-element techniques are used together with expansion, finite-element representation can either be used to represent the final result[27], or even be involved in the random walk[42].

The latter case may correspond either to the random-walk solution of the linear equation derived by projecting the integral equation, or to the Monte-Carlo evaluation of the multi-dimensional integral containing both the transport and the projection operators. The second case is preferred, because it does not require matrix $\mathbf{F}$ to be explicitly computed and stored.

The main problem of finite-element representations is that they require a lot of basis functions to accurately approximate high-variation, high-dimensional functions. Not surprisingly, finite-element methods become really popular only for the diffuse case, where the radiance depends on 2 scalars and is relatively smooth. For solving the non-diffuse case, they are good only if the surfaces are not very specular.

3.3. Diffuse versus the general case

If the surfaces have only diffuse reflection and emission — which is a general assumption of the *radiosity method*[12] — then the rendering (or the potential) equation has a simplified form:

$$L(\vec{x}) = L^e(\vec{x}) + \int_\Omega L(h(\vec{x}, -\omega'))\cdot f_r(\vec{x})\cdot\cos\theta'\, d\omega'. \tag{37}$$

In this case, the BRDF and the radiance depend on the surface point, but not on the direction, which reduces the inherent dimensionality of the problem and simplifies the finite-element representation:

$$L(\vec{x},\omega) \approx L^{(n)}(\vec{x}) = \sum_{j=1}^{n} L_j\cdot b_j(\vec{x}). \tag{38}$$

A widely used approach is the application of piece-wise constant basis functions for which $b_j(\vec{x}) = 1$ if $\vec{x}$ is on surface element A_j and 0 otherwise. An appropriate adjoint basis is $b_j(\vec{x}) = 1/A_j$ if $\vec{x}$ is on surface element A_j and 0 otherwise.

Using this set of basis functions, the original rendering equation is projected to the following linear equation:

$$\mathbf{L} = \mathbf{L}^e + \mathbf{F}\cdot\mathbf{L} \tag{39}$$

where

$$\mathbf{F}_{ij} = \langle \mathcal{T} b_j, \tilde{b}_i\rangle =$$

$$\int_S\int_\Omega b_j(h(\vec{x}, -\omega'))\cdot f_r(\vec{x})\cdot\cos\theta'\, d\omega'\tilde{b}_i(\vec{x})\, d\vec{x}. \tag{40}$$

Let us extend the formula of the solid angle to be valid for cases when $\vec{x}$ and $\vec{y}$ are not necessarily visible from each other. If the visibility indicator is $v(\vec{x},\vec{y})$, then

$$b_j(h(\vec{x}, -\omega'))\cdot d\omega' = \frac{d\vec{y}\cdot\cos\theta}{||\vec{x}-\vec{y}||^2}\cdot v(\vec{x},\vec{y}).$$

Using this substitution we obtain

$$\mathbf{F}_{ij} = \int_S\int_S b_j(\vec{y})\cdot\tilde{b}_i(\vec{x})\cdot f_r(\vec{x})\cdot\frac{\cos\theta'\cdot\cos\theta}{||\vec{x}-\vec{y}||^2}\cdot v(\vec{x},\vec{y})\, d\vec{y}\, d\vec{x}. \tag{41}$$

Taking advantage that the base functions are zero except for their specific domain, we get

$$\mathbf{F}_{ij} = \frac{f_i}{A_i}\cdot\int_{A_i}\int_\Omega b_j(h(\vec{x}, -\omega'))\cdot\cos\theta'\, d\omega'\, d\vec{x} =$$

$$\frac{f_i}{A_i}\cdot\int_{A_i}\int_{A_j}\frac{\cos\theta'\cdot\cos\theta}{||\vec{x}-\vec{y}||^2}\cdot v(\vec{x},\vec{y})\, d\vec{y}\, d\vec{x}. \tag{42}$$

Applying the

$$b_i(h(\vec{y},\omega))\cdot d\omega = \frac{d\vec{x}\cdot\cos\theta'}{||\vec{x}-\vec{y}||^2}\cdot v(\vec{x},\vec{y}).$$

substitution, we can derive yet another form of the transport matrix

$$\mathbf{F}_{ij} = \frac{f_i}{A_i}\cdot\int_{A_j}\int_\Omega b_i(h(\vec{y},\omega))\cdot\cos\theta\, d\omega\, d\vec{y}. \tag{43}$$

For the diffuse case, the adjoint equation can be derived as a special case of equation (35). Let W_i^e be 1 in points of A_i and at the directions of the hemisphere of A_i.

The *power equation* is then

$$\mathbf{P} = \mathbf{P}^e + \mathbf{H}\cdot\mathbf{P}, \tag{44}$$

where

$$\mathbf{H}_{ij} = \mathbf{F}_{ij}\cdot A_i/A_j = \mathbf{F}_{ji}\cdot f_i/f_j. \tag{45}$$

Note that using equation (43), we can also obtain

$$\mathbf{H}_{ij} = \frac{f_i}{A_j}\cdot\int_{A_j}\int_\Omega b_i(h(\vec{y},\omega))\cdot\cos\theta\, d\omega\, d\vec{y}. \tag{46}$$

© Institute of Computer Graphics 1998

In order to solve the projected integral equation, basically the same techniques can be applied as for the original integral equation: inversion, expansion and iteration.

3.3.1. Random walk solution of the projected integral equation

Expansion expands the solution into a discrete Neumann series

$$\mathbf{L} = \mathbf{L}^e + \mathbf{F} \cdot \mathbf{L}^e + \mathbf{F}^2 \cdot \mathbf{L}^e + \mathbf{F}^3 \cdot \mathbf{L}^e + \dots \qquad (47)$$

Let us again examine the $\mathbf{F}^2 \cdot \mathbf{L}^e$ term. Using the definition of the matrix $\mathbf{F}$, this can also be expressed as a multi-dimensional integral:

$$(\mathbf{F}^2 \cdot \mathbf{L}^e)|_i = \sum_{j=1}^{n} \sum_{k=1}^{n} \mathbf{F}_{ij} \cdot \mathbf{F}_{jk} \cdot \mathbf{L}^e_k =$$

$$\int_S \int_\Omega \int_S \int_\Omega \tilde{b}_i(\vec{x}_1) \cdot w_1(i) \cdot \sum_{j=1}^{n} b_j(h(\vec{x}_1, -\omega_1')) \cdot$$

$$\tilde{b}_j(\vec{x}_2) \cdot w_2(j) \cdot \sum_{k=1}^{n} b_k(h(\vec{x}_2, -\omega_2')) \cdot \mathbf{L}^e_k \, d\omega_2' d\vec{x}_2 d\omega_1' d\vec{x}_1,$$

where

$$\begin{aligned} w_1(i) &= f_i \cdot \cos\theta_1', \\ w_2(j) &= f_j \cdot \cos\theta_2'. \end{aligned} \qquad (48)$$

Considering the integrand, $\vec{x}_1$ should be in patch i for $\tilde{b}_i$ to be non zero. Then, only a single b_j will give non-zero value for the $\vec{y}_1 = h(\vec{x}_1, -\omega_1')$ point. To select this, a ray has to be traced from $\vec{x}_1$ in direction $-\omega_1'$ and the visible patch should be identified. Following this, another point on the identified patch i should be selected, which is denoted by $\vec{x}_2$, and a ray is traced in direction $-\omega_2'$ to obtain an index k of a patch whose emission should be propagated back on the walk. During propagation, the emission is multiplied by the BRDFs (f_i, f_j) and the cosine ($\cos\theta_2', \cos\theta_1'$) factors of the visited patches (figure 7).

Note that this is basically the same walking scheme, as used to solve the original integral equation. The fundamental difference is that when a patch is hit by the ray, the walk is not continued from the found point but from another point of the patch.

The power equation can be treated similarly. Again, let us examine the two-bounce case

$$(\mathbf{H}^2 \cdot \mathbf{P}^e)|_i = \sum_{j=1}^{n} \sum_{k=1}^{n} \mathbf{F}_{ji} \cdot \frac{f_i}{f_j} \cdot \mathbf{F}_{kj} \cdot \frac{f_j}{f_k} \cdot \mathbf{P}^e_k =$$

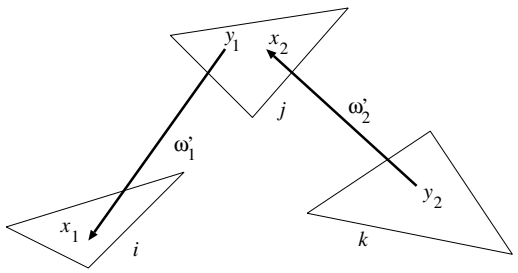

Figure 7: *Random walk solution of linear equation*

$$\int_S \int_\Omega \int_S \int_\Omega \mathbf{P}^e_k \cdot \tilde{b}_k(\vec{y}_1) \cdot w_1(k) \cdot \sum_{j=1}^{n} b_j(h(\vec{y}_1, \omega_1)) \cdot$$

$$\tilde{b}_j(\vec{y}_2) \cdot w_2(j) \cdot \sum_{k=1}^{n} b_i(h(\vec{y}_2, \omega_2)) \cdot w_3(i) \, d\omega_2 d\vec{y}_2 d\omega_1 d\vec{y}_1,$$

where

$$\begin{aligned} w_1(k) &= \cos\theta_1, \\ w_2(j) &= f_j \cdot \cos\theta_2, \\ w_3(i) &= f_i. \end{aligned} \qquad (49)$$

It means that the integrand in a single point can be obtained by selecting a point $\vec{y}_1$ on patch k, then tracing a ray in direction ω_1. Having identified the intersected patch j a new point $\vec{y}_2$ is selected on this patch and the ray-tracing is continued at direction ω_2. The patch which is hit by this ray receives the power of patch k attenuated by the BRDFs and the cosine factors of the steps.

3.4. Global versus local methods

Randomized transport operators transfer the radiance or the potential in the scene. The source and destination of the transfer can be points in the case of continuous methods or patches in the case of finite-element methods.

If the random operator is such that it always selects a single source for shooting or single destination for gathering, then the method is called *local method*. On the other hand, if many sources and destinations are taken into consideration simultaneously in each transfer, then the method is called *global method* or *multi-path method*[47].

Since global methods handle larger transfers in a single step, they can be expected to be more efficient than local methods. On the other hand, the single source or destination points of local methods directly correspond to the single "eye" of classical visibility algorithms. Thus, to exploit the capabilities of global methods, classical visibility algorithms should also be generalized for "moving" eye positions. These algorithms are called *global visibility algorithms*[43].

© Institute of Computer Graphics 1998

4. Stochastic expansion: random walks

In computer graphics the first Monte-Carlo random walk algorithm — called *distributed ray-tracing* — was proposed by Cook et al. [13], which spawned to a set of variations, including *path tracing*[26], *light-tracing*[17], *bi-directional path tracing*[30, 77], *Monte-Carlo radiosity*[54, 37, 42], and *two-pass methods* which combine radiosity and ray-tracing [52, 84, 79].

The problem of naive generation of walks is that the probability that a shooting path finds the eye is zero for a pin-hole camera or very small if a non-zero aperture camera model is used, while the probability that a gathering random path ends in a lightsource may be very little if the lightsources are small, thus the majority of the paths do not contribute to the image at all, and their computation is simply waste of time. Note that shooting is always superior for view-independent algorithms since they do not have to face the problem of small aperture.

Thus, on the one hand, random walk must be combined with a deterministic step that forces the walk to go to the eye and to find a lightsource. On the other hand, *importance sampling*[60] should be incorporated to prefer useful paths along which significant radiance is transferred. Note that although the contribution on the image is a function of the complete path, computer graphics applications usually assign estimated importance to individual steps of this path, which might be quite inaccurate. In a single step the importance is usually selected according to the BRDF [17, 30], or according to the direction of the direct lightsources [56]. Combined methods that find the important directions using both the BRDF and the incident illumination have been proposed in [76, 22, 31, 64]. Just recently, Veach and Guibas[78] proposed the Metropolis method to be used in the solution of the rendering equation. Unlike other approaches, Metropolis sampling[35] can assign importance to a complete walk not just to the steps of this walk, and it explores important regions of the domain adaptively while running the algorithm. Thus no a-priori knowledge is required about the important rays to construct a probability density function in advance. Instead, the algorithm converges to this probability density automatically.

4.1. Handling infinite-dimensional integrals

Expansion methods require the evaluation of infinite-dimensional integrals. One way of attacking the problem is truncating the Neumann series, but this introduces some bias, which can be quite high if the scene is highly reflective.

Fortunately, there is another approach that solves the infinite-dimensional integration problem through randomization. In the context of Monte-Carlo integration, this approach is called the *Russian roulette*[2], but here a somewhat more general treatment is given that can also justify this approach for quasi-Monte Carlo quadratures.

The basic idea is very simple. Higher order terms are included in the quadrature only randomly with probability decreasing with the order of the term. In order to compensate the missing terms in the expected value, the computed terms are multiplied by an appropriate factor. If the used probability goes to zero quickly, then the possibility of requiring very high dimensional integrals is rather low, which saves computation time but increases the variance. However, the expected value will still be correct, thus the integral quadrature will provide an asymptotically unbiased estimate.

A term of the Neumann series has generally the following form

$$I_n = \int \dots \int W(\mathbf{z}_1, \dots \mathbf{z}_n) \cdot L^e(\mathbf{z}_1, \dots \mathbf{z}_n)\, d\mathbf{z}_1 \dots \mathbf{z}_n, \tag{50}$$

where $W(\mathbf{z}_1, \dots \mathbf{z}_n) = w_0 \cdot w_1 \cdot \dots \cdot w_n$ is the product of the weights including the cosine functions of the angles and the BRDFs.

Let us randomize this integral by introducing a random variable $C(\mathbf{z}_1, \dots \mathbf{z}_n)$, called the *contribution indicator*, that is 1 if a sample $\mathbf{z}_1, \dots \mathbf{z}_n$ should be taken into account in the integral quadrature and 0 if it should not. Using this, we can define the following random variable,

$$I_n^* = \int \dots \int C \cdot \tilde{W} \cdot \tilde{L}^e\, d\mathbf{z}_1 \dots \mathbf{z}_n, \tag{51}$$

where $\tilde{W}$ and $\tilde{L}^e$ are appropriate modifications of W and L^e, which can compensate the missing terms.

The expectation value of this random variable is

$$E[I_n^*] = \int \dots \int E[C(\mathbf{z}_1, \dots \mathbf{z}_n)] \cdot \tilde{W} \cdot \tilde{L}^e\, d\mathbf{z}_1 \dots \mathbf{z}_n =$$

$$\int \dots \int p(\mathbf{z}_1, \dots \mathbf{z}_n) \cdot \tilde{W} \cdot \tilde{L}^e\, d\mathbf{z}_1 \dots \mathbf{z}_n, \tag{52}$$

where $p(\mathbf{z}_1, \dots \mathbf{z}_n)$ is the probability of using sample $\mathbf{z}_1, \dots \mathbf{z}_n$ in the integral quadrature.

Obviously, this equals to the original integral I if

$$p(\mathbf{z}_1, \dots \mathbf{z}_n) \cdot \tilde{W} \cdot \tilde{L}^e = W \cdot L^e. \tag{53}$$

There are many possible selection of the contribution indicator and the $\tilde{W}$ and $\tilde{L}^e$ functions, that can satisfy this requirement, thus there are many different unbiased estimators.

A widely used selection is letting

$$\tilde{W} = 1, \quad \tilde{L}^e = L^e \text{ and } p(\mathbf{z}_1, \dots \mathbf{z}_n) = W(\mathbf{z}_1, \dots \mathbf{z}_n).$$

which corresponds to continuing the walk after step i with probability $w(\mathbf{z}_i)$.

© Institute of Computer Graphics 1998

4.2. Importance sampling

When solving the rendering equation, usually directional integrals (or surface integrals in other formulation) should be evaluated. Thus to allow the application of random or low-discrepancy point sets, the integration domain should be transformed to the unit cube or square.

For example, when dealing with directions, we have to find a mapping $\omega = T(\mathbf{z})$ that projects the unit square to the surface of the sphere (or hemisphere) and use the following integration rule

$$\int_{\Omega} f(\omega)\, d\omega = \int_{[0,1]^D} f(T^{-1}(\mathbf{z})) \cdot \left| \frac{dT^{-1}(\mathbf{z})}{d\mathbf{z}} \right| d\mathbf{z}, \quad (54)$$

where

$$\left| \frac{dT^{-1}(\mathbf{z})}{d\mathbf{z}} \right| = \frac{1}{t(\mathbf{z})}$$

is the Jacobi determinant of the inverse mapping.

If the Jacobi determinant is large, then a small portion of the unit square is mapped onto a large region. Thus sample points that are uniformly distributed in the unit square will be quite rare in these regions. Alternatively, where the Jacobi determinant is small, the sample points will be dense. Considering this, the meaning of $t(T^{-1}(\mathbf{z}))$ is the *density* of sample points in the neighborhood of $\omega = T^{-1}(\mathbf{z})$. This has an illustrative content for the random case. If $\mathbf{z}$ is uniformly distributed random variable, then the probability density of $\omega = T(\mathbf{z})$ will be $t(\mathbf{z})$.

The same conclusion can also be made in the context of pure Monte-Carlo integration assuming that the samples are not uniformly distributed in the domain but following a $p(\mathbf{z})$ probability density:

$$\int_{[0,1]^D} f(\mathbf{z})\, d\mathbf{z} = \int_{[0,1]^D} \frac{f(\mathbf{z})}{p(\mathbf{z})} \cdot p(\mathbf{z})\, d\mathbf{z} =$$

$$E\left[\frac{f(\mathbf{z})}{p(\mathbf{z})}\right] \approx \frac{1}{M} \cdot \sum_{i=1}^{M} \frac{f(\mathbf{z}_i)}{p(\mathbf{z}_i)} \quad (55)$$

The variance of this estimate is low if $f(\mathbf{z})/p(\mathbf{z})$ is flat, thus $p(\mathbf{z})$ should be, at least approximately, proportional to $f(\mathbf{z})$.

Mathematically, the solution of either the rendering or the potential equation for a given point $(\vec{x}, \omega)$ requires the evaluation of the following multi-dimensional integral

$$L(\vec{x}, \omega) = L^e + \mathcal{T}L^e + \mathcal{T}^2 L^e + \ldots =$$

$$\int \ldots \int L^e + \frac{w_1}{t_1} \cdot L^e + \frac{w_1}{t_1} \cdot \frac{w_2}{t_2} \cdot L^e + \ldots \, d\mathbf{z}_1 d\mathbf{z}_2 \ldots \quad (56)$$

which can be estimated using formula (26) by evaluating the integrand in sample points and averaging the results.

An important design decision of such an algorithm is the selection of mappings T_i. Using probabilistic approach, it means the determination of the probability densities of finding new directions during the walks.

Following the directions concluded from the Koksma-Hlawka inequality, the mappings should make the integrand flat — that is of low variation, or constant in the ideal case. It means that the probability of selecting a walk is proportional to its contribution.

Looking at formula (56), which is the single multi-dimensional solution of the rendering equation, this decision seems to be hard to made, since there are too many free parameters to control simultaneously. Fortunately, this solution can also be presented in the following recursive form:

$$L^e + \int \frac{w_1}{t_1} \cdot [L^e + \int \frac{w_2}{t_2} \cdot [L^e + \ldots] \ldots] \, d\mathbf{z}_1 d\mathbf{z}_2 \ldots \quad (57)$$

If we could ensure that each of the integrands of the form

$$\int \frac{w_i}{t_i} \cdot [L^e + \int \ldots] \, d\mathbf{z}_i$$

is constant (at least approximately), then the integrand of the single multi-dimensional integral will also be constant.

An optimal importance sampling strategy thus requires density t_i to be proportional to the product of the incoming illumination $L^e + \int \ldots$ and the cosine weighted BRDF w_i. Unfortunately, during random walks the incoming non-direct illumination is not known (the random walk is just being done to estimate it).

Thus, we have three alternatives. Information about the illumination in the space can be gathered in a preprocessing phase, then this information can be used to obtain probability densities for importance sampling. This is called the *global importance sampling*.

The second alternative is using the information gained during previous walks to approximate the illumination. This strategy is called *adaptive importance sampling*.

In the third alternative, the problem is simplified and the indirect illumination is not considered in importance sampling. When the directions are generated, we use only w_i depending on the local orientation, the BRDF and L^e representing the direct illumination of the actual point. This is called the *local importance sampling*.

It turns out that we have to encounter severe problems when we have to find a mapping which has density that is proportional to the product of the effects of the BRDF and the direct lighting. Consequently, local importance sampling strategies usually use only either w_i or L^e to identify important directions. The first alternative is called the *BRDF sampling*, while the second is called the *lightsource sampling*.

4.2.1. BRDF sampling

BRDF based importance sampling means that at step i the density t_i of the sample points is proportional to the weight

© Institute of Computer Graphics 1998

w_i, that is

$$t_i \propto w_i = f_r(\omega_{\rm in}, \vec{x}, \omega_{\rm out}) \cdot \cos\theta \qquad (58)$$

In gathering algorithms $\omega_{\rm out}$ is known, θ is the angle between $\omega_{\rm in}$ and the surface normal, and $\omega_{\rm in}$ should be determined. In shooting algorithms, on the other hand, $\omega_{\rm in}$ is known, θ is the angle between $\omega_{\rm out}$ and the surface normal, and $\omega_{\rm out}$ should be determined.

Due to the fact that t_i represents density (probability density for Monte-Carlo methods), its integral is 1. Thus for gathering walks, the ratio of proportionality in equation (58) is

$$\int w\, d\omega_{\rm in} = \int f_r(\omega_{\rm in}, \vec{x}, \omega_{\rm out}) \cdot \cos\theta_{\rm in}\, d\omega_{\rm in} = a(\vec{x}, \omega_{\rm out})$$

where $a(\vec{x}, \omega_{\rm out})$ is the *albedo* of the surface at point $\vec{x}$ in the outgoing direction. Similarly, the proportionality ratio for shooting walks is

$$\int w\, d\omega_{\rm out} = \int f_r(\omega_{\rm in}, \vec{x}, \omega_{\rm out}) \cdot \cos\theta_{\rm out}\, d\omega_{\rm out} = a(\vec{x}, \omega_{\rm in}).$$

Thus the weights w_i/t_i are the albedos at the visited points.

When combining this with Russian roulette of type $\tilde{W} = 1, \tilde{L}^e = L^e$, the probability of continuing the walk will be equal to the albedo. This can also be interpreted in the following way. When the next direction is sampled, we use a subcritical density w_i which does not integrate to 1 but to a value $a(\vec{x}, \omega)$ and with the "missing" probability $1 - a(\vec{x}, \omega)$ it is decided whether or not the walk is stopped.

4.2.2. Lightsource sampling

Lightsource sampling is used in *direct lightsource calculations* [56] and as a complementary sampling strategy to BRDF sampling in random walks.

Since in this case, the samples are selected from the lightsource instead of the directional sphere, the surface integral form of the transport operator is needed:

$$(\mathcal{T} L^e)(\vec{x}, \omega) =$$

$$\int_\Omega L^e(h(\vec{x}, -\omega'), \omega') \cdot f_r(\omega', \vec{x}, \omega) \cdot \cos\theta'\, d\omega' =$$

$$\int_S L^e(\vec{y}, \omega_{\vec{y}\to\vec{x}}) \cdot f_r(\omega_{\vec{y}\to\vec{x}}, \vec{x}, \omega) \cdot \frac{\cos\theta'_{\vec{x}} \cdot \cos\theta_{\vec{y}}}{||\vec{x} - \vec{y}||^2} \cdot v(\vec{x}, \vec{y})\, d\vec{y}, \qquad (59)$$

where $v(\vec{x}, \vec{y})$ is 1 if points $\vec{x}$ and $\vec{y}$ are not occluded from each other and 0 otherwise.

If the scene has a single, homogeneous lightsource, it is relatively small and is far from the considered point, then the integrand will be approximately constant on the lightsource surface, thus point $\vec{y}$ can be generated using a uniform distribution on the lightsource.

If the scene has many lightsources, either one ray is sent to each of them, or just a single lightsource is sampled that is selected randomly.

4.2.3. Sampling the lightsources in gathering random walks

Since lightsource sampling generates samples only on the direct lightsources, it completely ignores indirect illumination. Thus it cannot be used alone in global illumination algorithms, but only as a complementary part of, for example, BRDF sampling.

The simplest way to combine the two strategies is to generate all but the last directions of the gathering walk by sampling the BRDF and to compute the last direction by sampling the lightsource. Note that when stopping the walk, the indirect illumination is assumed to be zero, thus following the directions of the lightsources is a reasonable approach.

Another combination strategy is to trace one or more shadow rays from each visited point of the walk towards the lightsources, not only from the last of them.

Formally, this approach can be presented as a restructuring of the Neumann series

$$L = L^e + \mathcal{T} L^e + \mathcal{T}^2 L^e + \mathcal{T}^3 L^e \ldots =$$

$$L^e + (\mathcal{T} L^e) + \mathcal{T}(\mathcal{T} L^e) + \mathcal{T}^2(\mathcal{T} L^e) \ldots \qquad (60)$$

and using lightsource sampling for the $L^{e*} = (\mathcal{T} L^e)$ integral while sampling the BRDFs when evaluating the $\mathcal{T}^i L^{e*}$ integrals. Practically it means that having hit a surface, one or more shadow rays are traced towards the lightsources and the reflection of the illumination of this point is estimated. This reflection is used as if it were the emission of the surface. This method is particularly efficient if the scene consists of point lightsources. Tracing a single ray to each point lightsource, the illumination due to the point lightsources can be determined exactly (with zero variance).

4.2.4. Importance sampling in colored scenes

So far, we have assumed that the weights containing the BRDFs and the emissions are scalars thus the densities can be made proportional to them. This is only true if the rendering equation is solved on a single wavelength.

However, if color images are needed, the rendering equation should be solved on several (at least 3) different wavelengths. If the different wavelengths are handled completely independently, then the proposed importance sampling strategy can be used without any modifications. However, this approach carries out geometric calculations, such as tracing rays, independently and redundantly for different wavelengths, thus it cannot be recommended.

© Institute of Computer Graphics 1998

A better approach is using rays that transport light on all wavelengths simultaneously. In this case the emission and the BRDF can be represented by vectors, thus to allow importance sampling, we need a scalar *importance function* $\mathcal{I}$ that is large when the elements in the vector are large and small when the elements are small. A straightforward way is using the *luminance* of the light as the importance function.

4.2.5. Multiple importance sampling

So far, we mentioned two basic importance sampling strategies, the BRDF sampling and the lightsource sampling, which are local in the sense that they focus on a single reflection. It is easy to imagine that if the sampling considers simultaneously many reflections, then the number of possible strategies increases dramatically.

Obviously, we desire to use the best sampling strategy. Unfortunately the performance of a sampling strategy depends on the properties of the scene, which is usually not known a-priori, thus the best strategy cannot be selected. Instead of selecting the best, Veach and Guibas[77] proposed to combine several strategies in a way that the strengths of the individual sampling methods are preserved.

Suppose that we can use n different sampling techniques for generating random paths, where the distribution of the samples is constructed from several $p_1, ..., p_n$ importance sampling distributions. The number of samples taken from p_i is denoted by M_i, and the total number of samples by $M = \sum_i M_i$. The M_i values are fixed in advance before any samples are taken. The "average probability density" of selecting the sample $\mathbf{z}$ is then

$$\hat{p}(\mathbf{z}) = \sum_{i=1}^{n} \frac{M_i}{M} \cdot p_i(\mathbf{z}). \tag{61}$$

Thus the integral quadrature using these samples is

$$\int_{[0,1]^D} f(\mathbf{z})\, d\mathbf{z} = \int_{[0,1]^D} \frac{f(\mathbf{z})}{\hat{p}(\mathbf{z})} \cdot \hat{p}(\mathbf{z})\, d\mathbf{z} \approx$$

$$\frac{1}{M} \sum_{i=1}^{n} \sum_{j=1}^{M_i} \frac{f(\mathbf{z}_{i,j})}{\hat{p}(\mathbf{z}_{i,j})} = \sum_{i=1}^{n} \frac{1}{M_i} \sum_{j=1}^{M_i} w_i(\mathbf{z}_{i,j}) \cdot \frac{f(\mathbf{z}_{i,j})}{p_i(\mathbf{z}_{i,j})} \tag{62}$$

where $\mathbf{z}_{i,j}$ is the jth sample taken from the ith distribution, and the weights are

$$w_i(\mathbf{z}) = \frac{M_i \cdot p_i(\mathbf{z})}{\sum_{k=1}^{n} M_k \cdot p_k(\mathbf{z})}. \tag{63}$$

Other heuristic weight factors can also provide good results[30, 75]. For the unbiased estimation $\sum_i w_i(\mathbf{z}) = 1$ should hold for all $\mathbf{z}$.

4.2.6. Global importance sampling

Global importance sampling methods are two-phase procedures. In a preprocessing phase they build a data structure that guides the second phase to find important directions. These methods can be classified according to their incorporated data structure. Since the ray-space is 5-dimensional, it is straightforward to apply a *5D adaptive tree*[31] that is similar to the well-known octree to store radiance information. Jensen proposed the application of the *photon-map* as the basis of importance sampling[22]. We assigned the power computed in the preprocessing phase to *links* established between two interacting patches[64].

4.2.7. Adaptive importance sampling

Adaptive importance sampling methods neither require the non-uniform probability densities to be constructed in advance, nor simplify them to take only into account local properties, but converge to a desired probability density using the knowledge of previous samples. Three techniques are particularly important, which have also been used in rendering: *genetic algorithms*[32] the Metropolis sampling[35, 78] and the VEGAS method[33, 62]. In this paper only the Metropolis sampling is discussed.

4.2.8. Metropolis sampling

The Metropolis algorithm[35] converges to the optimal probability density that is proportional to the importance, that is in the limiting case $\mathcal{I}(\mathbf{z}) = b \cdot p(\mathbf{z})$.

However, this probability density cannot be stored, thus in the Monte-Carlo formula the importance should be used instead, in the following way:

$$I = \int_V \frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})} \cdot \mathcal{I}(\mathbf{z})\, d\mathbf{z} = b \cdot \int_V \frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})} \cdot p(\mathbf{z})\, d\mathbf{z} =$$

$$b \cdot E\left[\frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})}\right] \approx \frac{b}{M} \cdot \sum_{i=1}^{M} \frac{f(\mathbf{z}_i)}{\mathcal{I}(\mathbf{z}_i)} \tag{64}$$

In order to generate samples according to $p(\mathbf{z}) = 1/b \cdot \mathcal{I}(\mathbf{z})$, a Markovian process is constructed whose stationary distribution is just $p(\mathbf{z})$. Informally, the next state $\mathbf{z}_{i+1}$ of this process is found by letting an almost arbitrary *tentative transition function* $T(\mathbf{z}_i \rightarrow \mathbf{z}_t)$ generate a *tentative sample* $\mathbf{z}_t$ which is either accepted as the real next state or rejected making the next state equal to the actual state using an "*acceptance probability*" $a(\mathbf{z}_i \rightarrow \mathbf{z}_t)$ that expresses the increase of the importance (if this "acceptance probability" is greater that 1, then the sample is accepted deterministically).

The formal definition of this Markovian process $\{\mathbf{z}_1, \mathbf{z}_2, \ldots \mathbf{z}_i \ldots\}$ is as follows:

© Institute of Computer Graphics 1998

for $i = 1$ **to** M **do**
  Based on the actual state $\mathbf{z}_i$,
    choose another random, tentative point $\mathbf{z}_t$
  $a(\mathbf{z}_i \to \mathbf{z}_t) = (\mathcal{I}(\mathbf{z}_t) \cdot T(\mathbf{z}_t \to \mathbf{z}_i))/((\mathcal{I}(\mathbf{z}_i) \cdot T(\mathbf{z}_i \to \mathbf{z}_t))$
  if $a(\mathbf{z}_i \to \mathbf{z}_t) \geq 1$ **then** accept $(\mathbf{z}_{i+1} = \mathbf{z}_t)$
  else *// accept with probability* $a(\mathbf{z}_i \to \mathbf{z}_t)$
    Generate random number r in $[0, 1]$.
    if $r < a(\mathbf{z}_i \to \mathbf{z}_t)$ **then** $\mathbf{z}_{i+1} = \mathbf{z}_t$
    else $\mathbf{z}_{i+1} = \mathbf{z}_i$
  endif
endfor

Note that "acceptance probability" $a(\mathbf{x} \to \mathbf{y})$ has the following property: $a(\mathbf{x} \to \mathbf{y}) = 1/a(\mathbf{y} \to \mathbf{x})$.

The transition probability of this Markovian process is:

$$P(\mathbf{x} \to \mathbf{y}) = \begin{cases} T(\mathbf{x} \to \mathbf{y}) & \text{if } a(\mathbf{x} \to \mathbf{y}) \geq 1\ , \\ T(\mathbf{x} \to \mathbf{y}) \cdot a(\mathbf{x} \to \mathbf{y}) & \text{otherwise.} \end{cases} \tag{65}$$

In equilibrium state, the transitions between two states $\mathbf{x}$ and $\mathbf{y}$ are balanced, that is

$$p(\mathbf{x}) \cdot P(\mathbf{x} \to \mathbf{y}) = p(\mathbf{y}) \cdot P(\mathbf{y} \to \mathbf{x}).$$

Using this and equation (65), and assuming without the loss of generality that $a(\mathbf{x} \to \mathbf{y}) \geq 1$, we can prove that the stationary probability distribution is really proportional to the importance:

$$\frac{p(\mathbf{x})}{p(\mathbf{y})} = \frac{P(\mathbf{y} \to \mathbf{x})}{P(\mathbf{x} \to \mathbf{y})} = \frac{T(\mathbf{y} \to \mathbf{x})}{T(\mathbf{x} \to \mathbf{y})} \cdot a(\mathbf{y} \to \mathbf{x}) = \frac{\mathcal{I}(\mathbf{x})}{\mathcal{I}(\mathbf{y})}. \tag{66}$$

If we select initial points according to the stationary distribution — that is proportionally to the importance — then the points visited in the walks originated at these starting points can be readily used in equation (64).

When we use Metropolis sampling in the solution of the global illumination problem, the "state" $\mathbf{z}$ corresponds to a complete walk. Mutation strategies are responsible for changing the walk a "little", by perturbing one or more directions or surface points, adding or deleting steps in the path, etc.

The first use of Metropolis sampling in rendering aimed at speeding up bi-directional path tracing[78].

4.3. Gathering-type random walk algorithms

Gathering type random walks correspond to the Monte-Carlo solution of the rendering equations. They start at the eye position and gather the emission of the visited points. This approach is quite ineffective if the lightsources are small, since it has rather low probability that a walk visits a lightsource.

4.3.1. Visibility ray-tracing

Classical ray tracing is a deterministic algorithm and is included here only for completeness. It only models ideal reflections and transmissions (also called the *coherent components*) that follow the laws of geometric optics — i.e. the law of reflection and the Snellius-Descartes law of refraction — but does not take into account multiple diffuse or incoherent specular reflection or refraction.

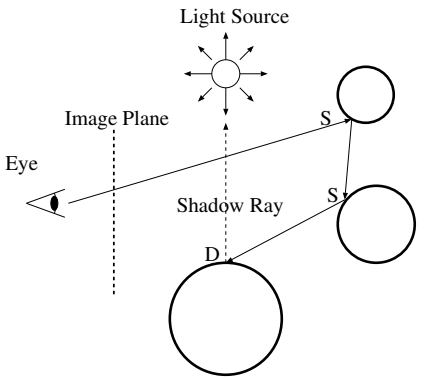

Figure 8: *Visibility ray tracing*

In visibility ray tracing (backward tracing) rays are emitted from the viewpoint. Rays are traced until they reach a surface which have no coherent reflection or refraction (unless they leave the environment or the length of the walk exceeds a predefined limit), so child rays are only generated (and traced recursively) when the given ray hits a surface that is reflective or transmissive. Restricting the continuation to coherent components, a ray can spawn to maximum two child rays, which is low enough not to necessitate random techniques.

The diffuse and incoherent specular reflection of a surface, on the other hand, is determined by taking into account only the direct illumination. For point lightsources, this can be done deterministically by tracing one ray, called *shadow ray*, to each lightsource to decide whether or not the given lightsource is visible from the intersection point. For area lightsources, the illumination can be computed by tracing random shadow rays as proposed by direct-lightsource computation.

4.3.2. Distributed ray-tracing

Distributed ray tracing suggested by Cook[13] can model all the possible paths. In this method the ray tracing is not terminated when reaching a diffuse surface. After a ray has hit a diffuse surface, child rays are generated randomly according to the BRDF characterizing the surface. For the appropriate estimation of the diffuse interreflection, child rays have to be traced and the average of their contributions have to be computed.

This approach is based on the recursive formulation of the integrals in the Neumann series (equation (18)).

© Institute of Computer Graphics 1998

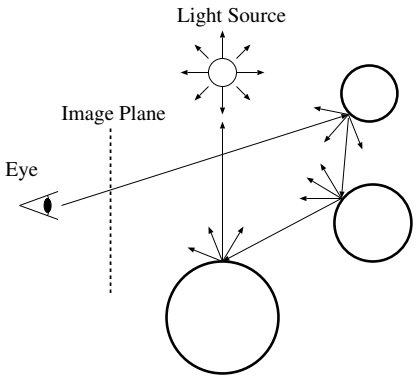

Figure 9: *Distributed ray-tracing*

4.3.3. Path-tracing

Another Monte-Carlo approach proposed by Kajiya is *path tracing* [26], which is based on the multi-dimensional integral formulation of the terms of the Neumann series (equation (15)).

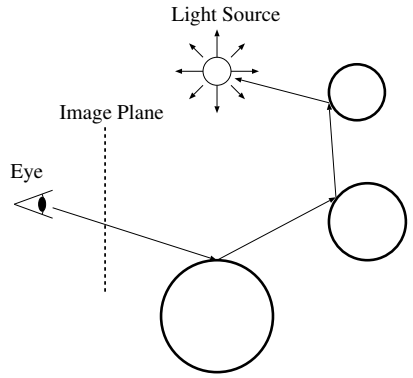

Figure 10: *Path tracing*

This method simply creates a path history for a single particle interacting with the environment until absorption. That is, rather than spawning new rays at an intersection, it simply chooses a random direction according to the BRDF for the ray to follow. The walk is continued with a probability equal to the albedo.

4.4. Shooting-type walks methods

Shooting walks are based on the Monte-Carlo solution of the potential equation.

4.4.1. Photon tracing

Photon tracing (forward ray-tracing) is the inverse of visibility ray-tracing and uses similar simplifying assumptions, thus they also stop tracing when hitting a surface that does not have coherent reflection or refraction. In photon tracing the rays are emitted from the light sources, and at each hit it is examined whether the surface has ideal reflection, refraction and incoherent reflection or refraction. In the directions of ideal reflection or refraction, the tracing is continued by starting new child rays. The effect of incoherent interactions, on the other hand, is stored in a map or is projected to the eye by tracing a ray towards the camera position.

4.4.2. Light Tracing

In *light tracing*[17] photons perform random walk through the scene starting at the lightsources. Whenever a surface is hit, a ray is traced from the intersection point to the eye and the contribution is added to the selected pixel (if any).

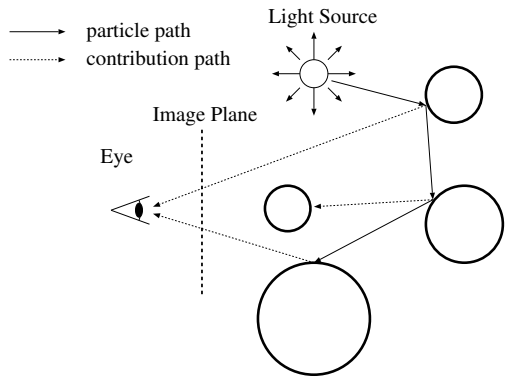

Figure 11: *Light tracing*

Light tracing is the direct implementation of the Monte-Carlo quadrature of the multi-dimensional formulation. When the next direction is determined, the BRDF based importance sampling can be applied and combined with the random termination according to the albedo.

4.4.3. Bi-directional Path Tracing

Bi-directional path tracing[30, 77] is based on the combination of shooting and gathering walks thus it can combine the advantages of both techniques. Namely, it can effectively handle small lightsources and small aperture cameras.

Walks are initiated at the same time from a selected light source and from the viewpoint. After some steps, either a single deterministic shadow ray is used to connect the two types of walks[77], or all points of the gathering walk are connected to all points of the shooting walk using deterministic rays[30]. If the deterministic shadow ray detects that the two points are occluded from each other, then the contribution of this path is zero.

Note that gathering and shooting walks use different integration variables, namely a gathering walk is specified by

© Institute of Computer Graphics 1998

a point on the pixel area and a sequence of incoming directions, while a shooting walk is defined by a point on the lightsource and a sequence of the outgoing directions. Thus when the two walks are connected, appropriate transformations should take place.

Let us first consider a walk of a single bounce (figure 12). According to the definition of the solid angle, we obtain

$$\frac{d\omega_1'}{d\omega_2} = \frac{dA \cdot \cos\theta_{\rm out}/r_1^2}{dA \cdot \cos\theta_{\rm in}/r_2^2} = \frac{r_2^2}{r_1^2} \cdot \frac{\cos\theta_{\rm out}}{\cos\theta_{\rm in}}, \qquad (67)$$

and for the substitution of the surface integral on the lightsource

$$d\omega_2' = \frac{d\vec{y} \cdot \cos\theta}{r_2^2}. \qquad (68)$$

Thus the transformation rule is

$$\cos\theta_1' \cdot \cos\theta_{\rm in}\; d\omega_1' d\omega_2' = \frac{\cos\theta_1' \cdot \cos\theta_{\rm out}}{r_1^2} \cdot \cos\theta\; d\omega_2 d\vec{y},$$

which means that when converting a shooting type walk to a gathering type walk, then the radiance should be multiplied by

$$\frac{\cos\theta_1' \cdot \cos\theta_{\rm out}}{r_1^2}.$$

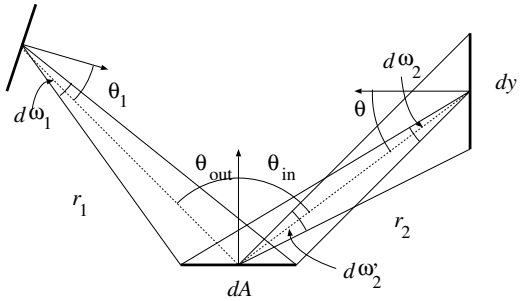

Figure 12: *Correspondence between the solid angles of incoming and outgoing directions*

When the shooting walk consists of more than 1 steps, then formula (67) should be applied to each of them, but formula (68) only to the last step. This conversion replaces the incoming directions by the outgoing directions and the subsequent steps compensate r_{k+1}^2/r_k^2 scaling. Finally, we end up with a formula which is similar to the 1-step case:

$$\cos\theta_k' \cdot \cos\theta_{k+1}' \cdot \ldots \cos\theta_n'\; d\omega_k' \ldots d\omega_n' =$$

$$\frac{\cos\theta_k' \cdot \cos\theta_{n-k+1}}{r_k^2} \cdot \cos\theta_{n-k} \cdot \ldots \cos\theta_1\; d\omega_{n-k} \ldots d\omega_1 d\vec{y}.$$

Figure 13 shows an example when $k = 2$ and $n = 4$. This formula means that we can use the rules of sections 1.2.1 and 1.2.2 to generate the shooting and gathering walks — gathering walks use the cosine of the incoming angle, while shooting walks use the cosine of the outgoing angle — and the transformation of the combined walk to a single gathering walk requires a multiplication by

$$\frac{\cos\theta_k' \cdot \cos\theta_{n-k+1}}{r_k^2}.$$

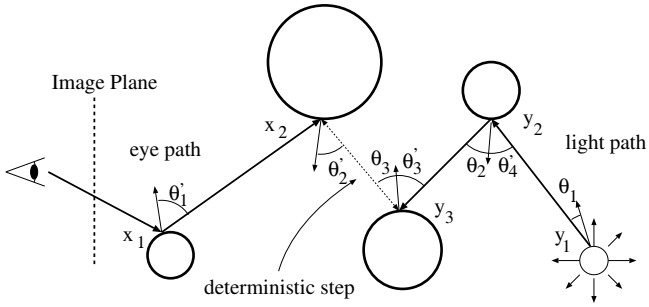

Figure 13: *Bi-directional path tracing with a single deterministic step*

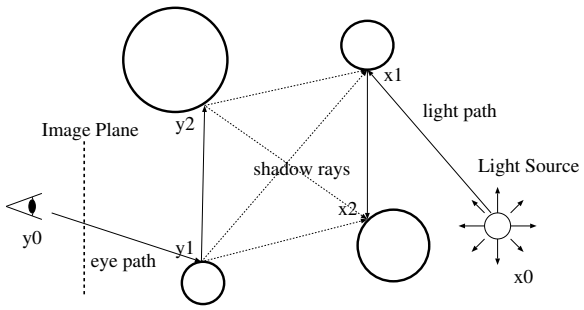

Figure 14: *Bi-directional path tracing with multiple deterministic steps*

In Lafortune's version of the bi-directional path tracing[30] not only the endpoints of the shooting and gathering walks are connected, but all intersection points are linked by shadow rays. The flux is estimated by a weighted sum of the different walks as suggested by the concept of multiple importance sampling.

4.4.4. Photon-map

Bi-directional path tracing connects a single gathering walk to a single shooting walk. However, if the effects of a shooting walk, for instance, could be stored, then when a new gathering walk is computed, it could be connected to all of them simultaneously. This is exactly what Jensen[24, 23, 25] proposed, also giving the definition of a data structure, called the *photon-map* which can efficiently store the effects of many shooting walks.

A photon map is a collection of photon hits generated in

© Institute of Computer Graphics 1998

the shooting phase of the algorithm. The photon-map is organized in a *kd-tree* to support efficient retrieval. A photon hit is stored with the power of the photon on different wavelengths, position, direction of arrival and with the surface normal.

The gathering phase is based on the following approximation of the transport operator:

$$L(\vec{x},\omega') = \int_{\Omega} L(h(\vec{x},-\omega'),\omega')\cdot f_r(\omega',\vec{x},\omega)\cdot\cos\theta'\,d\omega' =$$

$$\int_{\Omega} \frac{d\Phi(\omega')}{dA\cos\theta' d\omega'}\cdot f_r(\omega',\vec{x},\omega)\cdot\cos\theta'\,d\omega' \approx$$

$$\sum_{i=1}^{n} \frac{\Delta\Phi(\omega_i')}{\Delta A}\cdot f_r(\omega_i',\vec{x},\omega), \tag{69}$$

where $\Delta\Phi(\omega_i')$ is the power of a photon landing at the surface ΔA from direction ω_i'. The $\Delta\Phi$ and ΔA quantities are approximated from the photons in the neighborhood of $\vec{x}$ in the following way. A sphere centered around $\vec{x}$ is extended until it contains n photons. If at this point the radius of the sphere is r, then the intersected surface area is $\Delta A = \pi r^2$.

4.4.5. Instant radiosity

Instant radiosity[28] elegantly subdivides the shooting walks into a view-independent walk and into the projection of the contribution to the eye. Let us call this last step with eye projection the *eye-step*. The view-independent walk is quite similar to the light-tracing algorithm, but the new directions are sampled from the Halton sequence instead of a random distribution.

When a surface hit is found, the eye-step is calculated taking advantage of the rendering hardware of advanced workstations. The reflection of this hit is assumed to be a point lightsource (in the radiosity setting the emission of the lightsource is also diffuse), and the rendering hardware is used to render the effect of this lightsource on the scene and also to compute shadows. The final image is the average of such estimates, which are computed using the hardware accumulation buffer.

Instant radiosity is quite similar to photon-map based techniques. However, instead of using ray-tracing for final gather, the photons in the photon map are used as lightsources and fast and hardware supported visibility and shadow algorithms are applied. The other fundamental difference is that instant radiosity allows just a relatively low number of photons which therefore should be very well distributed. The optimal distribution is provided by quasi-Monte Carlo light walks.

4.4.6. Random walks for the radiosity setting

As mentioned, the projected rendering equation can also be solved by random walks[54, 47]. The basic difference is that when a patch is hit by a ray, then instead of initiating the next ray from this point, another independent point is selected on the same patch.

Considering the concept of importance sampling and Russian roulette, many different strategies can be elaborated by appropriately defining the p, $\tilde{W}$ and $\tilde{L}^e$ functions (recall that according to equation (53) the requirement of an unbiased estimate is $p\cdot\tilde{W}\cdot\tilde{L}^e = W\cdot L^e$).

For example, let us use the following simulation[54, 47] to obtain a radiance estimate of patch i_1:

First a ray is found that starts on this patch. The starting point $\vec{x}_1$ is sampled from a uniform distribution, while the direction ω_1' is sampled from a cosine distribution, thus the probability density is $1/A_{i_1}\cdot\cos\theta_1'/\pi$. This ray is traced and the next patch is identified. Let it be patch i_2. At patch i_2 it is decided whether or not the walk should be stopped with probability of the albedo of the patch. Note that for diffuse surfaces the albedo is $a = f\cdot\pi$. If the walk has to be continued, then a new starting point $\vec{x}_2$ is found on patch i_2, and the same procedure is repeated recursively.

With this strategy, the probability density of completing an n step walk is

$$p(\vec{x}_1,\omega_1',\vec{x}_2,\omega_2',\ldots\vec{x}_{n-1},\omega_{n-1}') =$$

$$\frac{1}{A_{i_1}}\cdot\frac{\cos\theta_1'}{\pi}\cdot\frac{a_{i_2}}{A_{i_2}}\cdot\frac{\cos\theta_2'}{\pi}\cdots\frac{a_{i_{n-1}}}{A_{i_{n-1}}}\cdot\frac{\cos\theta_{n-1}'}{\pi}\cdot(1-a_{i_n}) =$$

$$\frac{f_{i_1}}{A_{i_1}}\cdot\cos\theta_1'\cdot\frac{f_{i_2}}{A_{i_2}}\cdot\cos\theta_2'\cdots\frac{f_{i_{n-1}}}{A_{i_{n-1}}}\cdot\cos\theta_{n-1}'\cdot\frac{1-a_{i_n}}{a_{i_1}} =$$

$$W\cdot\frac{1-a_{i_n}}{a_{i_1}}. \tag{70}$$

Thus the required weight $\tilde{W}$ of the walk is

$$\tilde{W} = \frac{a_{i_1}}{1-a_{i_n}}. \tag{71}$$

Thus if the patch on which the walk is terminated is a source having emission $\mathbf{L}_n^e$, then the estimator of the radiance of patch i is

$$\mathbf{L}_n^e\cdot\frac{a_{i_1}}{1-a_{i_n}}.$$

Other gathering or shooting estimators have been proposed and their variances have been estimated in [54, 47].

4.4.7. Global ray-bundle tracing

Realizing that an accurate solution requires great many samples, *global ray-bundle tracing*[68, 69, 62] uses a bundle of very many (e.g. 1 million or even infinite) global parallel rays,

© Institute of Computer Graphics 1998

which can be traced simultaneously using image coherence techniques. In order to represent the radiance that is transferred by a ray, finite-element techniques are applied that approximate the positional (but not the directional) dependence of the radiance by piece-wise continuous or piece-wise linear functions[67].

$$L(\vec{x},\omega) \approx \sum_{j=1}^{n} b_j(\vec{x}) \cdot L_j(\omega) = \mathbf{b}^T \cdot \mathbf{L}(\omega). \qquad (72)$$

Note that this is a mixed finite-element and continuous method, since the positional dependence of the radiance is approximated by finite-elements, while the directional dependence is not.

Substituting this into the rendering equation and projecting that into an adjoint base we obtain

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \mathcal{T}_F \mathbf{L}(\omega), \qquad (73)$$

where $\mathcal{T}_F$ is a composition of the original transport operator and its projection to the adjoint base

$$\mathcal{T}_F \mathbf{L}(\omega) = \langle \mathcal{T}\mathbf{b}^T \cdot \mathbf{L}(\omega), \tilde{\mathbf{b}} \rangle. \qquad (74)$$

Let us use again piece-wise constant basis functions. Then the result of the application of the transport operator on patch i is

$$\mathcal{T}_F \mathbf{L}(\omega)|_i = \frac{1}{A_i} \cdot \int_\Omega \int_{A_i} L(h(\vec{x},-\omega'),\omega') \cdot \cos\theta' \cdot \tilde{f}_i(\omega',\omega)\, d\vec{x}\, d\omega'. \qquad (75)$$

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

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

$$\sum_{j=1}^{n} \tilde{f}_i(\omega',\omega) \cdot A(i,j,\omega') \cdot L_j(\omega'), \qquad (76)$$

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 15.

This projected area can be efficiently calculated simultaneously for all patch pairs using global discrete or continuous visibility algorithms[62] and also exploiting the hardware z-buffer[69]. These algorithms can also have random nature, that is, they can result in $A(i,j,\omega') \cdot L_j(\omega')$ just as an the expected value[63].

Using equation (76) the rendering equation can be obtained as:

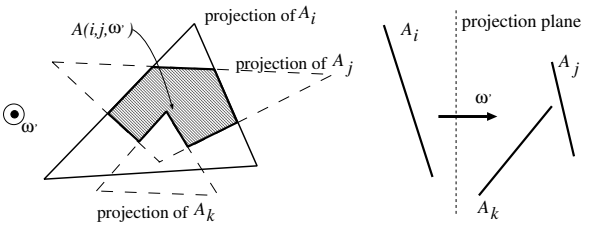

Figure 15: *Interpretation of* $A(i,j,\omega')$

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \int_\Omega \mathbf{F}(\omega',\omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, d\omega', \quad (77)$$

where $\mathbf{L}(\omega)$ is the vector of radiance values, $\mathbf{F}(\omega',\omega)$ is a diagonal matrix of BRDFs, and *geometry matrix* $\mathbf{A}$ contains the relative visible areas: $\mathbf{A}(\omega')|_{ij} = A(i,j,\omega')/A_i$.

Note that equation (77) is highly intuitive as well. The radiance of a patch is the sum of the emission and the reflection of all incoming radiance. The role of the patch-direction-patch "form-factors" is played by $A(i,j,\omega')/A_i$.

This is also an integral equation but unlike the original rendering equation it provides the radiance of not only a single point but for all points at once. This integral equation is solved by random or quasi-random shooting type walks.

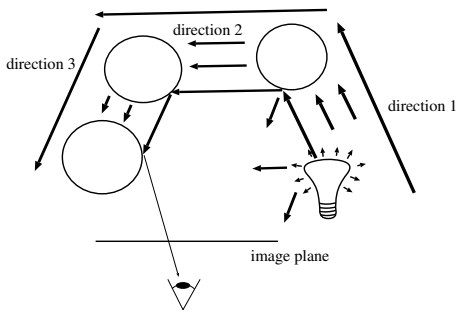

Figure 16: *A path of ray-bundles*

A single walk starts by selecting a direction either randomly or quasi-randomly, and the emission transfer of all patches is calculated into this direction (figure 16). Then a new direction is found, and the emission is transferred and the incoming radiance generated by the previous transfer is reflected from all patches into this new direction. The algorithm keeps doing this for a few times depending on how many bounces should be considered, then the emission is sent and the incoming radiance caused by the last transfer is reflected towards the eye. Averaging these contributions results in the final 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

© Institute of Computer Graphics 1998

is assigned to them, then in order to avoid "blocky" appearance, bi-linear smoothing 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.

In the simplest case, when the global ray-bundles are handled similarly to the light-tracing algorithm, this requires just one variable per patch. In the more elaborate versions of the algorithm, the radiance of the steps of a walk are combined in all possible ways, which significantly improves the performance but needs more storage space.

The global directions defining the ray-bundles are sampled from uniform random and low-discrepancy sequences. In [69, 62] adaptive importance sampling, such as the Metropolis method, is also considered, but it is concluded that this method does not offer significant improvement for relatively smooth integrands.

4.4.8. Multi-path method using global random lines

Multi-path methods represent a bridge between random walk and iterative methods. They are essentially random walk methods, but in their single step many random walks are advanced parallely.

Sbert[47, 50, 49] proposed a complete family of multi-path methods, that are based on random *global lines*, which was 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 intersected 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.

4.5. Software implementations of the random-walk methods

An important practical implementation of the path-tracing algorithm is the *RADIANCE* system[80]. It incorporates a lot of coherence, importance sampling and caching techniques to efficiently handle complex scenes. For example, the scene is stored in an octree to reduce ray-object intersections. The irradiance is cached in another octree[81] to take advantage of the fact that diffuse reflections do not change abruptly. This allows to compute the color of neighboring pixels without tracing rays in the directions responsible for diffuse interreflections.

Another versatile package is the *ART*[72] developed at the Vienna University of Technology. It is an open system that can easily be extended by new global illumination algorithms, BRDF models, object types, etc. ART has an efficient ray-casting engine using hierarchical bounding boxes to reduce ray-object intersection calculations, a scene processing subsystem that takes generalized CSG models as scene descriptions, and an image postprocessing subsystem to handle different image formats. Visibility ray-tracing, path tracing and photon tracing have already been implemented. These methods use BRDF based importance sampling and terminate the walks using Russian roulette. Bi-directional path tracing with Metropolis sampling is being implemented. The system can work with R, G, B coordinates and also with different spectral representations.

5. 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 was proposed for the diffuse radiosity problem in [37], that is for the solution of finite-dimensional linear equations.

In this section we present a generalized formulation that is somewhat different from the original concepts to allow to attack also non-diffuse global illumination problems[63].

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

$$E[\mathcal{T}^* L] = \mathcal{T} L \tag{78}$$

for any integrable function L.

In the case of finite-element representations, equation (78) should be true for the $\mathcal{T}_F \mathbf{L}$ operator that also involves the projection to the finite function space.

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 used in the iteration formula:

$$L_n = L^e + \mathcal{T}_n^* L_{n-1}. \tag{79}$$

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

1. Random "point" p_i is found from a finite dimensional set Π using probability density $\mathrm{prob}(p)$. 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 L.

© Institute of Computer Graphics 1998

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

$$P_n = \mathcal{M}L_n \tag{80}$$

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 iteration 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:

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

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

$$\mathcal{M}(L^e + \frac{1}{M}\sum_{i=1}^{M} \mathcal{T}_i^* L^e + \frac{M-1}{M} \cdot \frac{1}{M-1}\sum_{i=1}^{M-1} \mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e + \ldots). \tag{81}$$

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 (78). For $k = 1$, the *total expectation value theorem* can be applied:

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e] = \int_{\Pi} E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] \cdot \mathrm{prob}(p)\, dp. \tag{82}$$

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:

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] = \mathcal{T}_{i+1}^*(p)\,(E[\mathcal{T}_i^* L^e])\,. \tag{83}$$

Using requirement (78) for the expected value we further obtain

$$E[\mathcal{T}_{i+1}^* \mathcal{T}_i^* L^e | p_{i+1} = p] = \mathcal{T}_{i+1}^*(p)(\mathcal{T} L^e). \tag{84}$$

Substituting this back to equation (82), we get

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

$$E[\mathcal{T}_{i+1}^*(\mathcal{T} L^e)] = \mathcal{T}(\mathcal{T} L^e) = \mathcal{T}^2 L^e. \tag{85}$$

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 is then

$$E[\tilde{P}] =$$

$$\mathcal{M}(L^e + \mathcal{T}L^e + \frac{M-1}{M}\mathcal{T}^2 L^e + \frac{M-2}{M}\mathcal{T}^3 L^e + \ldots + \frac{1}{M}\mathcal{T}^M L^e), \tag{86}$$

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 upperbounded by[63]

$$\frac{1}{M} \cdot \frac{\lambda^2}{(1-\lambda)^2} \cdot ||L^e||.$$

Another error formula is presented for the diffuse case in [47]. This can be neglected for high number of iterations, or can even be reduced by ignoring the first few iterations in the averaged result [37, 47].

Finally, it must be explained why random variable $\tilde{P}$ converges to its expected value. Looking at formula (81) 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[46] 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.

5.1. Other averaging techniques

In the previous section we solved the problem that stochastic iteration is not convergent by simply averaging the values generated during iteration. There are other averaging schemes, however, that use even more combinations of the

© Institute of Computer Graphics 1998

preceding random operators. In the subsequent sections two such schemes are presented.

5.1.1. Semi-iteration

Semi-iteration[37] uses the following formulae to derive a new value from the previous one:

$$\begin{aligned} L'_n &= L^e + \mathcal{T}_n^* L_{n-1}, \\ L_n &= \tau_n \cdot L'_n + (1-\tau_n) \cdot L_{n-1}, \\ \tilde{P}_n &= \mathcal{M} L_n, \end{aligned} \tag{87}$$

where τ_n is an appropriate sequence that converges to 0, as for example, $\tau_n = 1/n$.

To allow comparison, the corresponding formulae of the normal iteration are also presented here:

$$\begin{aligned} L_n &= L^e + \mathcal{T}_n^* L_{n-1}, \\ \tilde{P}_n &= \tau_n \cdot \mathcal{M} L_n + (1-\tau_n) \cdot \tilde{P}_{n-1} \end{aligned} \tag{88}$$

Note that the fundamental difference is that semi-iteration uses the average of the previous samples not only in the final estimate but also to continue iteration. Semi-iteration thus can use all combinations of the preceding random operators to compute the actual result. However, it also has energy defect.

5.1.2. D-step iteration

Let us approach stochastic iteration from the direction of reducing the bias of finite-length random walks. The bias can be eliminated using a simple correction of the emission function L^e when calculating higher order interreflections.

Note that a global walk of length D provides the following terms:

$$L^e + \mathcal{T}_1^* L^e + \mathcal{T}_{(1,2)}^* L^e \ldots + \mathcal{T}_{(1,D)}^* L^e,$$

where

$$\mathcal{T}_{(i,j)}^* = \mathcal{T}_j^* \mathcal{T}_{j-1}^* \ldots \mathcal{T}_{i+1}^* \mathcal{T}_i^*.$$

Thus having computed the first walk, we also have an estimate for $\mathcal{T}_{(1,D)}^* L^e = \mathcal{T}_D^* \mathcal{T}_{D-1}^* \ldots \mathcal{T}_2^* \mathcal{T}_1^* L^e$. Let us use this estimate to correct the emission function in the higher order terms when the second walk is computed:

$$L^e + \mathcal{T}_{D+1}^*(L^e + \mathcal{T}_{(1,D)}^* L^e) + \ldots + \mathcal{T}_{(D+1,2D)}^*(L^e + \mathcal{T}_{(1,D)}^* L^e) =$$

$$L^e + \mathcal{T}_{D+1}^* L^e + \ldots + \mathcal{T}_{(D+1,2D)}^* L^e +$$

$$\mathcal{T}_{(1,D+1)}^* L^e + \ldots + \mathcal{T}_{(1,2D)}^* L^e. \tag{89}$$

This gives us estimates not only for the bounces from 0 to D but also for the bounces from $D+1$ to $2D$. Again the last-bounce will store $\mathcal{T}_{(1,D)}^* L^e + \mathcal{T}_{(1,2D)}^* L^e$, which can be used to compensate the emission. Thus after the second step we have estimates for the 0 to $3D$ bounces. Asymptotically, this method will generate estimates for all bounces. However, if M global walks are generated, then the number of estimates for bounces of 0 to D is M, for bounces of $D+1$ to $2D$ is $M-1$, for bounces $2D+1$ to $3D$ is $M-2$ etc., which still results in some small energy defect.

This type of iteration takes D steps before making an iteration step, which allows the combination of the steps in more sophisticated ways. Such a combination happens in bi-directional path-tracing using multiple deterministic steps[30] and also in global ray-bundle tracing[62].

5.2. Definition of random transport operators

In order to use this general stochastic iteration scheme in practice, the key problem is the definition of the random transport operator. This operator should meet the requirement of equation (78) and should be easy to compute.

For the continuous case, a single application of the transport operator contains a directional integral. For the finite element case, the transport operator also includes the projection to the adjoint basis which requires additional integration in the domain of basis functions. This additional integration means a surface integral for the diffuse radiosity setting and also for the ray-bundle tracing. For other non-diffuse finite-element methods a surface and a directional integrals need to be evaluated (note that directional integrals are sometimes "hidden" by integrals on the surfaces visible at different directions).

Following the general concepts of Monte-Carlo methods, we usually do not intend to compute the integrals *explicitly*, but want to get them as an expected value. Thus different random transport operators can be classified according to which integrals are evaluated explicitly using some deterministic quadrature and which integrals are computed *implicitly* as an expectation value.

5.3. Transport operator for the continuous, non-diffuse setting

The continuous formulation has just a single directional integral, thus a random transport operator can evaluate this single integral implicitly. This results in a method that uses a "single" random walk to obtain the solution.

An example of such single walk techniques is the following modification of the light tracing algorithm[63]:

In each step i a ray is obtained that has random origin $\vec{y}_i$ and direction ω_i with a probability that is proportional to the cosine weighted radiance of this point at the given direction. This ray is traced and the whole power

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

is transported to that point $\vec{x}$ which is hit by the ray. Formally

© Institute of Computer Graphics 1998

the random transport operator is

$$(\mathcal{T}^* L)(\vec{x},\omega) = \Phi \cdot \delta(\vec{x} - h(\vec{y},\omega_i)) \cdot f_r(\omega_i,\vec{x},\omega). \quad (90)$$

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. This probability depends on the albedo $a_{\vec{x}_i}(\omega_i)$ of the found point.

The algorithm selects initially 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*[2, 57].

5.4. Random transport operators for the diffuse radiosity

In the gathering type radiosity algorithms the projected transport operator has the following form

$$\mathcal{T}_F \mathbf{L} = \mathbf{F} \cdot \mathbf{L}.$$

Alternatively, shooting radiosity algorithms are based on the projected potential equation $\mathcal{T}_F' \mathbf{P} = \mathbf{H} \cdot \mathbf{P}$.

According to the basic requirement of stochastic iteration we need to find random operators $\mathcal{T}_F^*$ or $\mathcal{T}_F'^*$ that behave as the real operator in average, that is

$$E[\mathcal{T}_F^* \mathbf{L}] = \mathbf{F} \cdot \mathbf{L}, \quad (91)$$

$$E[\mathcal{T}_F'^* \mathbf{P}] = \mathbf{H} \cdot \mathbf{P}. \quad (92)$$

The evaluation of $(\mathbf{F} \cdot \mathbf{L})_i$ or alternatively $(\mathbf{H} \cdot \mathbf{P})|_i$ requires a surface and a directional integration (or in other formulations two surface integrations).

The possible alternatives for a random transport operator are

1. Both integrals are explicitly computed but only for a randomly selected subset of the patches.
2. The surface integral explicitly computed but the directional integral implicitly.
3. Compute the surface integral implicitly but the directional integral explicitly. This method can, for example, use hemicubes for the directional integration but selects the center of the hemicube randomly on the patch.
4. Both integrals are computed implicitly.

5.4.1. Stochastic radiosity

In *stochastic radiosity*[38], the randomized operator is simplified in a sense that it first selects a single (or a few) patches with probability proportional to their power and then calculates the transfer only from this important patch as if it had all the power $\Phi = \sum_{k=1}^{n} \mathbf{P}_k$. Thus here both integrals are explicitly computed but only for a subset of patches.

To prove that it meets requirement stated by equation (92), let us examine the new power of patch i and suppose that patch j has been selected.

$$(\mathcal{T}_F'^* \mathbf{P})|_i = \mathbf{H}_{ij} \cdot \Phi \quad (93)$$

Since the probability of selecting patch j is $\mathbf{P}_j/\Phi$, the expectation of the new power is

$$E[(\mathcal{T}_F'^* \mathbf{P})|_i] = \sum_{j=1}^{n} \mathbf{H}_{ij} \cdot \Phi \cdot \frac{\mathbf{P}_j}{\Phi} = \sum_{j=1}^{n} \mathbf{H}_{ij} \cdot \mathbf{P}_j \quad (94)$$

which we wanted to prove.

5.4.2. Transillumination radiosity

The *transillumination radiosity method*[37, 66] has also a stochastic iteration version. It defines the random transport operator by uniformly selecting M *transillumination directions* $\omega_1', \ldots \omega_M'$ and allowing patches to interact only in these transillumination directions. In order to calculate these interactions, a large discretized window is placed perpendicularly to each transillumination direction and the radiance transfer to a patch is approximated by elementary transfers going through the pixels covering the projection of the patch.

Let us consider a single transillumination direction. Projecting patch A_i onto a plane that is perpendicular to the transillumination direction and then approximating the integral of the incoming radiance here by a discrete sum, we get

$$\int_{A_i} L(h(\vec{x},-\omega_d')) \cdot \cos\theta_d' \, d\vec{x} =$$

$$\int_{A_i^p} L(h(\vec{x}',-\omega_d')) \cdot \, d\vec{x}' \approx \sum_{P \in A_i^p} L_{\mathrm{buffer_d}[P]} \cdot \delta A. \quad (95)$$

where $\mathrm{buffer_d}[P]$ stores the index of that patch which is visible in pixel P in the transillumination direction ω_d' from patch i, and δA is the size of a pixel of the buffer (figure 17).

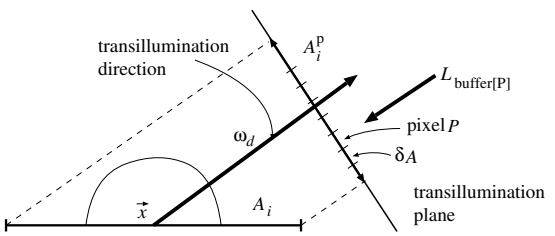

Figure 17: *Integration on the transillumination plane*

© Institute of Computer Graphics 1998

Thus the random transfer operator is

$$(\mathcal{T}_F^* \mathbf{L})|_i = \frac{4\pi \cdot f_i \cdot \delta A}{M} \sum_{d=1}^{M} \sum_{P \in A_i^p} L_{\mathrm{buffer_d}[P]}. \qquad (96)$$

If the transillumination directions are uniformly distributed and the buffer is uniformly jittered, then the expected value of this operator is

$$E[(\mathcal{T}_F^* \mathbf{L})|_i] = \frac{1}{M} \sum_{d=1}^{M} \int_\Omega \int_P \sum_{P \in A_i^p} L_{\mathrm{buffer_d}[P]} \cdot f_i \cdot \; dp \; d\omega'_d.$$

If uniform jittering is applied during rendering into the buffer, then we can usually assume that the discrete approximation of the positional radiance distribution gives back the real distribution in the average case, that is

$$\int_P \sum_{P \in A_i^p} L_{\mathrm{buffer_d}[P]} \; dp = \int_{A_i^p} L(h(\vec{x}', -\omega'_d)) \; d\vec{x}'. \qquad (97)$$

However, this statement is not always true if incremental polygon filling algorithms are applied[62, 63]. Note, for example, then an incremental polygon filling algorithm always generates an approximation whose width and height are at least 1. Thus this assumption is correct if the resolution of the discrete buffer is high enough to project each polygon onto at least a few pixels.

Substituting this to the expectation value integral we get

$$E[(\mathcal{T}_F^* \mathbf{L})|_i] = \frac{1}{M} \sum_{d=1}^{M} \int_\Omega \int_{A_i^p} L(h(\vec{x}', -\omega'_d)) \; d\vec{x}' \cdot f_i \; d\omega'_d =$$

$$\int_\Omega \int_{A_i} L(h(\vec{x}', -\omega')) \cdot f_i \cdot \cos\theta' \; d\vec{x} d\omega'. \qquad (98)$$

Using $L(h(\vec{x}', -\omega')) = \sum_{j=1}^{n} b_j(h(\vec{x}', -\omega')) \cdot \mathbf{L}_j$ and equation (42), we can prove that the expectation really gives back the real projected transport operator:

$$E[(\mathcal{T}_F^* \mathbf{L})|_i] =$$

$$\sum_{j=1}^{n} \int_\Omega \int_{A_i} b_j(h(\vec{x}', -\omega')) \cdot f_i \cdot \cos\theta'_d \; d\vec{x} d\omega' \cdot \mathbf{L}_j = \sum_{j=1}^{n} \mathbf{F}_{ij} \cdot \mathbf{L}_j. \qquad (99)$$

5.4.3. Stochastic ray-radiosity

Stochastic ray-radiosity[40] approximates the transport operator by M random rays that are sampled proportionally to the power of the patches. On a patch the starting point of the ray is sampled using a uniform distribution, while the direction follows a cosine distribution. A single ray carries Φ/M power. Thus this method approximates both integrals implicitly.

Let us examine the case when a single ray is selected (since different rays are sampled from the same distribution, the effect of M rays will be M times the effect of a single ray in the expected value). Suppose that patch j is selected as a shooting patch. The probability of the selection event is $\mathbf{P}_j/\Phi$. Thus the probability density of selecting a point $\vec{x}$ of a patch and a direction ω is

$$\frac{\mathbf{P}_j}{\Phi} \cdot \frac{1}{A_j} \cdot \cos\theta.$$

This transfers Φ/M power to the patch that is hit by the ray where the reflected power is computed. Thus the random transport operator for a single ray is

$$E[(\mathcal{T}_F^{'*} \mathbf{P})|_i] =$$

$$M \cdot f_i \cdot \sum_{j=1}^{n} \int_{A_j} \int_\Omega b_i(h(\vec{y}, \omega)) \cdot \frac{\Phi}{M} \cdot \frac{1}{A_j} \cdot \cos\theta \; d\vec{y} d\omega \cdot \frac{\mathbf{P}_j}{\Phi} =$$

$$\sum_{j=1}^{n} \frac{f_i}{A_j} \cdot \int_{A_j} \int_\Omega b_i(h(\vec{y}, \omega)) \cdot \cos\theta \; d\vec{y} d\omega \cdot \mathbf{P}_j = \sum_{j=1}^{n} \mathbf{H}_{ij} \cdot \mathbf{P}_j. \qquad (100)$$

5.5. Transport operators for the non-diffuse finite-element case

When moving towards the non-diffuse case, another requirement must be imposed upon the random transport operator. It must not only meet the requirement of equation (78), be easy to compute, but it must also allow the compact representation of the $\mathcal{T}_i^* L$ functions. This extra requirement is evident if we take into account that unlike in the diffuse case, 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). From the point of view of compact representation, what we have to avoid is 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 the domain of the radiance function and using random transport operators that require the

© Institute of Computer Graphics 1998

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 $p_{\rm eye}$ which is needed by the measuring device and for previous point providing $[\mathcal{T}^*(p_i)L(p_i)](p_{\rm eye})$.

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**
  $P^{\rm new} = L^e(p_{\rm eye}) + [\mathcal{T}^*(p_i)L(p_i)](p_{\rm eye})$
  $P = \mathcal{M}P^{\rm new} \cdot 1/i + (1 - 1/i) \cdot P$
  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

Note that using normal iteration we have to store the radiance L just in a single point p_i, while in semi-iteration all the previous points should be remembered. In semi-iteration the important feature that the transport operator should be evaluated just for a single point pair p_i, p_{i+1} is lost. In the case of D-step iteration, the computation needs to be done at a finite number of point pairs whose number is limited by $\binom{D}{2}$. For semi-iteration, however, there is no such upper limit, which eventually results in requiring the complete representation of the function. This can be allowed in diffuse case, but not in the general case, thus in methods handling the non-diffuse case normal iteration is preferred, D-step iteration is still allowed, but we have to avoid semi-iteration, despite of its better combination capability.

5.5.1. Global ray-bundle based iteration

Recall that the finite-element approximation applied by ray-bundle tracing converts the rendering equation to the following form (section 4.4.7):

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \mathcal{T}_F \mathbf{L}(\omega), \tag{101}$$

where $\mathcal{T}_F$ is a composition of the original transport operator and its projection to the adjoint base

$$\mathcal{T}_F \mathbf{L}(\omega) = \int_\Omega \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, d\omega'. \tag{102}$$

Let the random approximation of the transport operator be the transfer of the radiance of all surface points of the scene in a single uniformly distributed random direction. This transfer can be effectively realized by sending a ray-bundle into this direction. Thus the random transport operator is ω'

$$(\mathcal{T}^* \mathbf{L})(\omega) = 4\pi \cdot \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega'). \tag{103}$$

If the directions are sampled from a uniform distribution, then this obviously gives back the integral operator as an expected value:

$$E[(\mathcal{T}^* \mathbf{L})(\omega)] = \int_\Omega 4\pi \cdot \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, \frac{d\omega'}{4\pi} = \mathcal{T}_F \mathbf{L}(\omega). \tag{104}$$

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

The resulting algorithm is quite simple. In a step of the stochastic iteration a new direction is found and this direction together with the previous direction are used to evaluate the random transport operator. Then an image estimate is computed by reflecting the previously computed radiance estimate towards the eye. The complete algorithm is summarized in the following:

Generate the first random global direction ω_1
for each patch i **do** $L[i] = L_i^e(\omega_1)$
for $m = 1$ **to** M **do** *// iteration cycles*
  Calculate the image estimate relfecting
  the incoming radiance $L[1], L[2], \ldots L[n]$ from ω_m towards the eye
  Average the estimate with the Image
  Generate random global direction ω_{m+1}
  for each patch i **do**
    $L^{\rm new}[i] = L_i^e(\omega_{m+1}) +$
    $4\pi \cdot \sum_{j=1}^{n} \tilde{f}_i(\omega_m, \omega_{m+1}) \cdot A(i, j, \omega_m)/A_i \cdot L[j]$
  endfor
endfor
Display Image

The methods to calculate the reflection of the incoming radiance towards the eye are the same as in ray-bundle tracing.

© Institute of Computer Graphics 1998

5.6. 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 randomization 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*[29]. 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.

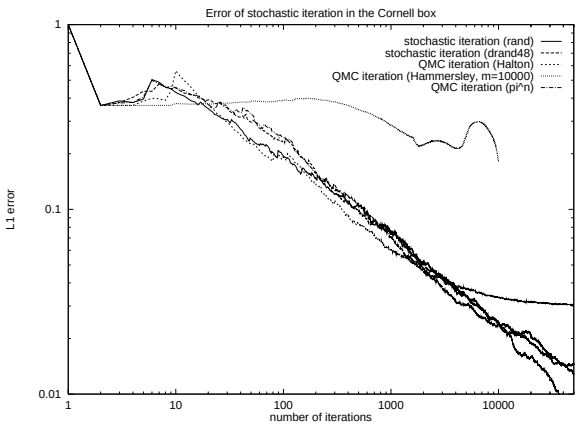

Figure 18: *Ray-bundle based stochastic iteration with random and quasi-random numbers*

In order to avoid this problem without getting rid of the quasi-Monte Carlo sequences, [66] proposed the random scrambling of the sample points. The same problem arises, for example, when generating uniform distributions on a sphere, for which [9] 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[29]).

In order 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 18 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[15]. However, this sequence is very unstable numerically, therefore we used the $q_n = \{(\pi - 2) \cdot q_{n-1} \bmod 100000\}$ scheme.

6. Initial smoothing: first shot

Monte-Carlo integration is efficient if the integrand is relatively smooth and does not exhibit high variations. For global methods, point lightsources may pose problems. Fortunately, these lightsources can be easily handled separately by deterministic techniques.

This algorithm can be applied in a preprocessing step, and is called the *first-shot*. For the diffuse radiosity problem, the preprocessing type first-shot algorithm has been first presented in [50], extended to multiple interreflections in [9] and has been generalized to non-diffuse environments in [69].

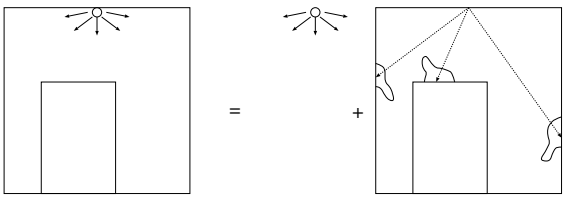

Figure 19: *First shot technique*

Formally, the unknown radiance L is decomposed into two terms:

$$L = L^{ep} + L^{np} \tag{105}$$

where L^{ep} is the emission of the small, point-like lightsources, L^{np} is the emission of the area lightsources and the reflected radiance. Substituting this into the rendering equation we have:

$$L^{ep} + L^{np} = L^e + \mathcal{T}(L^{ep} + L^{np}). \tag{106}$$

Introducing the new lightsource term

$$L^{e*} = L^e - L^{ep} + \mathcal{T}L^{ep} \tag{107}$$

which just replaces the point lightsources (L^{ep}) by their effect ($\mathcal{T}L^{ep}$), the equation for L^{np} is similar to the original rendering equation:

$$L^{np} = L^{e*} + \mathcal{T}L^{np}. \tag{108}$$

© Institute of Computer Graphics 1998

It means that first the direct illumination caused by the point lightsources must be computed, then they can be removed from the scene and added again at the end of the computation.

In diffuse environments the storage of the direction independent L^{e*} function requires just one extra variable per patch.

In non-diffuse environments, however, L^{e*} can be a non-constant function, which is difficult to represent and store. Instead, the incoming radiance received by the patches from each point lightsource should be stored (this requires l additional variables per patch, where l is the number of point lightsources). When L^{e*} is needed for a given direction, then it is computed on the fly from these incoming radiances.

7. Error and complexity of the stochastic global illumination methods

This paper reviewed a lot of different global illumination techniques. Thus at the end of the paper, a natural expectation would be a qualitative comparison of these methods, or set of criteria that tell for a particular scene which method is better than the others. However, ranking these methods is a very hard task.

Even the quality metrics are difficult to define. We can use, for example, error metrics and running time measurements, and say that if a methods gets more accurate results in shorter time than it is better than the other. The error metrics can be simple norms of the difference of the actual and a reference radiance function, or can even be based on *perceptual based metrics*[45]. Running time measurements depend also on the computer used. To formulate resource requirements of a particular algorithm in a machine independent way, complexity measures can be used.

Unfortunately, error measures and even abstract complexity measures are very difficult to derive analytically for a given algorithm. A taxonomy of different errors occurring when solving the rendering equation is presented in[3]. The error of the solution of the diffuse rendering equation was analyzed by [34]. An error metric to guide Russian roulette and splitting has been introduced by [7].

The complexity of classical radiosity algorithms has been investigated in [71] and concluded that deterministic, non-hierarchical algorithms require $O(n^2)$ time, where n is the number of patches. Stochastic methods seem to be better, since they can achieve $O(n \log n)$ time complexity [48, 43, 54, 65, 6].

In [66], we have provided a technique based on the transillumination method to reformulate the diffuse radiosity problem to make the integrand have finite variation, to fully take advantage of the improved performance provided by the low-discrepancy series. This reformulation also allowed to apply the Koksma-Hlawka inequality in analytic error formulae.

8. Conclusions

This paper presented a review of stochastic global illumination algorithms.

9. Acknowledgements

This work has been supported by the National Scientific Research Fund (OTKA), ref.No.: F 015884 and the Austrian-Hungarian Action Fund, ref.No.: 29p4, 32öu9 and 34öu28.

References

1. J. Arvo. Stratified sampling of spherical triangles. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 437–438, 1995.
2. J. Arvo and D. Kirk. Particle transport and image synthesis. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, pages 63–66, 1990.
3. J. Arvo, K. Torrance, and B. Smits. A framework for the analysis of error in global illumination algorithms. *Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 75–84, 1994.
4. L. Aupperle and P. Hanrahan. A hierarchical illumination algorithms for surfaces with glossy reflection. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 155–162, 1993.
5. G. Baranoski, R. Bramley, and P Shirley. Fast radiosity solutions for environments with high average reflectance. In *Rendering Techniques '96*, pages 345–355, 1996.
6. P. Bekaert, L. Neumann, A. Neumann, M. Sbert, and Y. Willems. Hierarchical Monte-Carlo radiosity. In *Rendering Techniques '98*, pages 259–268, 1998.
7. M. R. Bolin and G. W. Meyer. An error metric for Monte Carlo ray tracing. In *Rendering Techniques '97*, pages 57–68, 1997.
8. C. Buckalew and D. Fussell. Illumination networks: Fast realistic rendering with general reflectance functions. *Computer Graphics (SIGGRAPH '89 Proceedings)*, 23(3):89–98, July 1989.
9. 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.
10. 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.
11. 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.
12. Michael Cohen and Donald Greenberg. The hemi-cube, a radiosity solution for complex environments. In *Computer Graphics (SIGGRAPH '85 Proceedings)*, pages 31–40, 1985.
13. R. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, pages 137–145, 1984.

© Institute of Computer Graphics 1998

14. B. Csébfalvi. A review of Monte-Carlo ray tracing algorithms. In *CESCG '97, Central European Seminar on Computer Graphics*, pages 87–103, 1997.

15. I. Deák. *Random Number Generators and Simulation*. Akadémia Kiadó, Budapest, 1989.

16. D. P. Dobkin, D. Eppstein, and D. P. Mitchell. Computing the discrepancy with applications to supersampling patterns. *ACM Transactions on Graphics*, 15(4):354–376, 1996.

17. P. Dutre, E. Lafortune, and Y. D. Willems. Monte Carlo light tracing with direct computation of pixel intensities. In *Compugraphics '93*, pages 128–137, Alvor, 1993.

18. A. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers, Inc., San Francisco, 1995.

19. P. Hanrahan, D. Salzman, and L. Aupperle. Rapid hierachical radiosity algorithm. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 1991.

20. M. Hyben, I. Martisovits, and A. Ferko. Scene complexity for rendering in flatland. In L. Szirmay-Kalos, editor, *Spring Conference on Computer Graphics*, pages 112–120, 1998.

21. D. S. Immel, M. F. Cohen, and D. P. Greenberg. A radiosity method for non-diffuse environments. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, pages 133–142, 1986.

22. H. W. Jensen. Importance driven path tracing using the photon maps. In *Rendering Techniques '95*, pages 326–335, 1995.

23. H. W. Jensen. Global illumination using photon maps. In *Rendering Techniques '96*, pages 21–30, 1996.

24. H. W. Jensen and N. J. Christensen. Photon maps in bidirectional Monte Carlo ray tracing of complex objects. *Computers and Graphics*, 19(2):215–224, 1995.

25. H. W. Jensen and P. H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. *Computers and Graphics (SIGGRAPH '98 Proceedings)*, pages 311–320, 1998.

26. J. T. Kajiya. The rendering equation. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, pages 143–150, 1986.

27. A. Keller. A quasi-Monte Carlo algorithm for the global illumination in the radiosity setting. In H. Niederreiter and P. Shiue, editors, *Monte-Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 239–251. Springer, 1995.

28. A. Keller. Instant radiosity. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 49–55, 1997.

29. D.E. Knuth. *The art of computer programming. Volume 2 (Seminumerical algorithms)*. Addison-Wesley, Reading, Mass. USA, 1981.

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

31. 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.

32. B. Lange and B. Beyer. Rayvolution: An evolutionary ray tracing algorithm. In *Photorealistic Rendering Techniques*, pages 136–144, 1994.

33. G. P. Lepage. An adaptive multidimensional integration program. Technical Report CLNS-80/447, Cornell University, 1980.

34. D. Lischinski, B. Smits, and D.P. Greenberg. Bounds and error estimates for radiosity. In *Computer Graphics Proceedings, Annual Conference Series*, pages 67–75, 1994.

35. N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1091, 1953.

36. D. P. Mitchell. Consequences of stratified sampling in graphics. *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 277–280, 1996.

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

38. 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.

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

40. 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.

41. H. Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, Pennsilvania, 1992.

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

43. M. Pellegrini. Monte Carlo approximation of form factors with error bounded a priori. *Discrete and Computational Geometry*, 1997. (to appear).

44. William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C (Second Edition)*. Cambridge University Press, Cambridge, USA, 1992.

45. J. Prikryl and W. Purgathofer. Perceptually based radiosity. In *Eurographics '98, STAR — State of the Art Report*, 1998.

46. Alfréd Rényi. *Wahrscheinlichkeitsrechnung*. VEB Deutscher Verlag der Wissenschaften, Berlin, 1962.

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

48. M. Sbert. Error and complexity of random walk Monte-Carlo radiosity. *IEEE Transactions on Visualization and Computer Graphics*, 3(1), 1997.

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

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

© Institute of Computer Graphics 1998

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

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

53. P. Shirley. Discrepancy as a quality measure for sampling distributions. In *Eurographics '91*, pages 183–194. Elsevier Science Publishers, 1991.

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

55. P. Shirley. Monte Carlo simulation and integration. *SIGGRAPH '93 Global Illumination Course Notes (42)*, 1993.

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

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

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

59. F. X. Sillion, J. R. Arvo, S. H. Westin, and D. P. Greenberg. A global illumination solution for general reflectance distributions. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):187–198, 1991.

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

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

62. 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/.

63. 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/.

64. L. Szirmay-Kalos, B. Csébfalvi, and W. Purgathofer. Importance-driven quasi-Monte Carlo solution of the rendering equation. In *Winter School of Computer Graphics '98*, pages 377–386, Plzen, Czech Republic, 1998.

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

66. 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 (Eurographics'97)*, 16(3):271–281, 1997.

67. L. Szirmay-Kalos, T. Fóris, and W. Purgathofer. Non-diffuse, random-walk radiosity algorithm with linear basis functions. *Machine Graphics and Vision*, 7(1):475–484, 1998.

68. L. Szirmay-Kalos, T. Fóris, and W. Purgathofer. Quasi-Monte Carlo global ray-bundle tracing with infinite number of rays. In *Winter School of Computer Graphics '98*, pages 386–393, Plzen, Czech Republic, 1998.

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

70. L. Szirmay-Kalos and W. Purgathofer. Quasi-Monte Carlo solution of the rendering equation. Technical Report TR-186-2-98-22, Institute of Computer Graphics, Vienna University of Technology, 1998. www.cg.tuwien.ac.at/.

71. László Szirmay-Kalos and Gábor Márton. On convergence and complexity of radiosity algorithms. In *Winter School of Computer Graphics '95*, pages 313–322, Plzen, Czech Republic, 14–18 February 1995.

72. R. F. Tobler. ART — Advanced Rendering Toolkit. 1998. http://www.cg.tuwien.ac.at/ reseach/ rendering/ ART.

73. R. F. Tobler, L. Neumann, M. Sbert, and W. Purgathofer. A new form factor analogy and its application to stochastic global illumination algorithms. In *Rendering Techniques '98*, 1998.

74. R. F. Tobler, A. Wilkie, M. Feda, and W. Purgathofer. A hierarchical subdivision algorithm for stochastic radiosity methods. In *Rendering Techniques '97*, pages 193–203, 1996.

75. E. Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, http://graphics.stanford.edu/papers/veach_thesis, 1997.

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

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

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

79. 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 Proceedings)*, pages 311–324, 1987.

80. G. J. Ward. The RADIANCE lighting simulation and rendering system. *Computer Graphics*, 28(4):459–472, 1994.

81. G. J. Ward, F. M. Rubinstein, and R. D. Clear. A ray-tracing solution for diffuse interreflection. *Computer Graphics*, 22(4):85–92, 1988.

82. T. Warnock. Computational investigations of low-discrepancy point sets. In H. Niederreiter and P. Shiue, editors, *Monte-Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pages 354–361. Springer, 1995.

83. A. Wilkie, R. F. Tobler, and W. Purgathofer. Photon radiosity lightmaps for CSG solids. In *CSG '98 — Set theoretic Solid Modelling Techniques and Applications*, 1998.

84. 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.

© Institute of Computer Graphics 1998