Global Ray-bundle Tracing with Hardware Acceleration

László Szirmay-Kalos [1], Werner Purgathofer
Department of Control Engineering and Information Technology, TU of Budapest
Budapest, Műegyetem rkp. 11, H-1111, HUNGARY
szirmay@fsz.bme.hu

Abstract. The paper presents a single-pass, view-dependent method to solve the general rendering equation, using a combined finite element and random walk approach. Applying finite element techniques, the surfaces are decomposed into planar patches that are assumed to have position independent, but not direction independent radiance. The direction dependent radiance function is then computed by random walk using bundles of parallel rays. In a single step of the walk, the radiance transfer is evaluated exploiting the hardware z-buffer of workstations, making the calculation fast. The proposed method is particularly efficient for scenes including not very specular materials illuminated by large area light-sources or sky-light. In order to increase the speed for difficult lighting situations, walks can be selected according to their importance. The importance can be explored adaptively by the Metropolis sampling method.

1 Introduction

The fundamental task of computer graphics is to solve a Fredholm type integral equation describing the light transport. This equation is called the *rendering equation* and has the following form:

$$L(\vec{x},\omega) = L^e(\vec{x},\omega) + \int_\Omega L(h(\vec{x},-\omega'),\omega') \cdot \cos\theta' \cdot f_r(\vec{x},\omega',\omega)\, d\omega' \qquad (1)$$

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', θ' is the angle between the surface normal and direction $-\omega'$, and $f_r(\vec{x},\omega',\omega)$ is the bi-directional reflection/refraction function. Since the rendering equation contains the unknown radiance function both inside and outside the integral, in order to express the solution, this coupling should be resolved. Generally, two methods can be applied for this: finite element methods or random walk methods.

Finite element methods project the problem into a finite function base and approximate the solution here. The projection transforms the integral equation to a system of linear equations for which straightforward solution techniques are available. Finite element techniques that aim at the solution of the non-diffuse case can be traced back to the finite element approximation of the directional functions using *partitioned sphere*

[1]This work has been supported by the National Scientific Research Fund (OTKA), ref.No.: F 015884, the Austrian-Hungarian Action Fund, ref.No.: 29p4, 32öu9 and 34öu28, and the Spanish-Hungarian Fund, ref.No.: E9.

[8] or *spherical harmonics* [23], and to the application of *extended form factors* [22]. Since the radiance function is not smooth and is of 4-variate if non-diffuse reflection should also be considered, finite element methods require a great number of basis functions, and thus the system of linear equations will be very large. Although, hierarchical or multiresolution methods [4] and clustering [3] can help, the memory requirements are still prohibitive for complex scenes.

Random walk methods, on the other hand, resolve the coupling by expanding the integral equation into a Neumann series, and calculate the resulting high-dimensional integrals by numerical quadrature from discrete samples. A single discrete sample corresponds to a complete photon-path (called the *walk*) from a lightsource to the eye, which is usually built by d ray-shooting steps if the photon is reflected d times. Since classical quadrature rules are useless for the calculation of very high dimensional integrals, Monte-Carlo or quasi-Monte Carlo techniques must be applied.

In computer graphics the first Monte-Carlo random walk algorithm — called *distributed ray-tracing* — was proposed by Cook et al. [5], which spawned to a set of variations, including *path tracing* [9], *light-tracing* [7], *Monte-Carlo radiosity* [20][14][17], and two-pass methods which combine radiosity and ray-tracing [29].

The problem of naive generation of walks is that the majority of the paths do not contribute to the image at all, and their computation is simply waste of time.

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. *Light tracing* connects each bounce position to the eye deterministically [7]. *Bi-directional path-tracing* methods start a walk from the eye and a walk from a lightsource and connect the bounce positions of the two walks [11],[27].

On the other hand, importance sampling [24] 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 [7] [11], or according to the direction of the direct lightsources [21]. Combined methods that find the important directions using both the BRDF and the incident illumination have been proposed in [26], [12]. Just recently, Veach and Guibas [28] proposed the Metropolis method to be used in the solution of the rendering equation. Unlike other approaches, Metropolis sampling [13] 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.

In order to reduce the noise of these methods, very many samples are required, especially when importance sampling cannot help significantly — that is when the lightsources are large and the surfaces are not very specular. One way of reducing the ray-object intersection calculation cost is storing this information in the form of *illumination networks* [1], but it has large memory requirements, and representing the light-transport of small number of predefined rays might introduce artifacts.

The proposed new method also combines the advantages of finite-element and random-walk approaches and can solve the general non-diffuse case. The method needs no preprocessing, the memory requirements are modest, and it is particularly efficient for scenes containing larger area lightsources and moderately specular surfaces — that is where other importance-sampling walk methods become inefficient.

2 Informal discussion of the new algorithm

Walk methods proposed so far use individual ray-paths as samples of the integrand of the rendering equation. However, ray-shooting may waste a lot of computation by ignoring all the intersections but the one closest to the start of the ray. Thus it seems worth using a set of *global directions* [18][14] for the complete scene instead of solving the visibility problem independently for different points $\vec{x}$. Moreover, ray-shooting is a simple but by no means the most effective visibility algorithm since it is unable to take advantage of image coherence. Other methods based on the exploitation of image coherence, such as the z-buffer, painter's, Warnock's, etc. algorithms can be considered as handling a bundle of parallel rays and solving the visibility problem for all of them simultaneously. Continuous (also called object-precision) methods can even determine the visibility problem independently of the resolution, which corresponds to tracing infinitely many parallel rays simultaneously. This paper aims at using ray-bundles instead of individual rays to solve the rendering equation.

These visibility algorithms assume that the surfaces are decomposed into planar patches, thus the proposed method also uses this assumption. On the other hand, the patch decomposition is also used for the finite-element structure, where each surface element is assumed to have uniform radiance (note that this does not mean that the radiance function itself is uniform or diffuse, just the same radiance function is valid anywhere inside a patch). In order to simulate multiple interreflections, ray-bundles should be traced several times in different directions. The series of directions is called a *global random walk*.

2.1 Computation of global ray-bundle walks

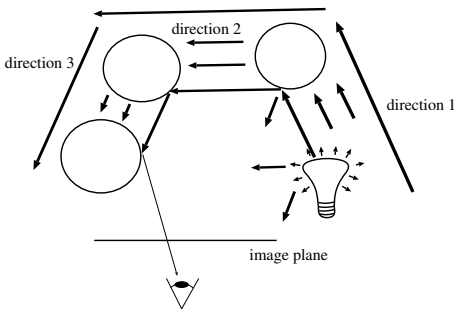

Fig. 1. A path of ray-bundles

The algorithm takes samples of these global walks and uses them in the quadrature. 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 1). Then a new direction is found, and the emission is transferred and the irradiance 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 irradiance caused by the last transfer is reflected towards the eye. Averaging these contributions results in the final image. When the radiance reflection is calculated, the radiance is attenuated by the BRDF of the corresponding surface element.

In order to make this method work, efficient algorithms are needed that can compute the radiance transfer of all patches in a single direction. A new global visibility method capable of taking advantage of the built-in z-buffer is proposed for this task.

In the following sections, first the formal aspects of the combination of finite element and random walk techniques are discussed, then the global visibility method is presented.

3 Reformulation of the rendering equation using the finite-elements

According to the concept of finite-elements, the radiance, emission and the BRDF of patch i are assumed to be constant and are denoted by $L_i(\omega)$, $L_i^e(\omega)$ and $\tilde{f}_i(\omega, \omega')$, respectively. Using these assumptions, integrating the rendering equation on patch i we obtain:

$$L_i(\omega) \cdot A_i = L_i^e(\omega) \cdot A_i + \int_{\Omega} \int_{A_i} L(h(\vec{x}, -\omega'), \omega') \cdot \cos\theta' \cdot \tilde{f}_i(\omega', \omega)\, d\vec{x}\, d\omega'. \quad (2)$$

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

$$\int_{A_i} L(h(\vec{x}, -\omega'), \omega') \cdot \cos\theta' \cdot \tilde{f}_i(\omega', \omega)\, d\vec{x} = \sum_{j=1}^{n} \tilde{f}_i(\omega', \omega) \cdot A(i, j, \omega') \cdot L_j(\omega'), \quad (3)$$

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 2.

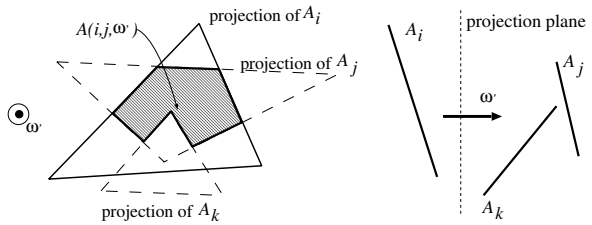

Fig. 2. Interpretation of $A(i, j, \omega')$

Using equation (3) and dividing both sides by A_i, the matrix form of the rendering equation can be obtained as:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \int_{\Omega} \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, d\omega', \quad (4)$$

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

3.1 Random walk

The solution of integral equation (4) can be obtained in the form of a Neumann series:

$$\mathbf{L}(\omega) = \sum_{i=0}^{\infty} \mathcal{T}^i \mathbf{L}^e(\omega), \quad \text{where} \quad \mathcal{T}\mathbf{L}(\omega) = \int_{\Omega} \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, d\omega'. \qquad (5)$$

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

In practice the infinite sum of the Neumann series is always approximated by a finite sum. The number of required terms is determined by the contraction of the operator $\mathcal{T}$ — that is the overall reflectivity of the scene. Let us denote the maximum number of calculated bounces by D. The truncation of the Neumann series introduces a bias in the estimation, which can be tolerated if D is high enough.

In order to simplify the notations, we introduce the *max d-bounce irradiance* $\mathbf{I}_d$ for $d = 1, 2, \ldots$ as follows:

$$\begin{aligned} \mathbf{I}_0 &= \mathbf{A}(\omega'_D) \cdot \mathbf{L}^e(\omega'_D) \\ \mathbf{I}_d &= \mathbf{A}(\omega'_{D-d}) \cdot \left(\mathbf{L}^e(\omega'_{D-d}) + 4\pi \cdot \mathbf{F}(\omega'_{D-d+1}, \omega'_{D-d}) \cdot \mathbf{I}_{d-1}\right), \end{aligned}$$

where $\mathbf{I}_d$ is a $d+1$ dimensional function of directions $(\omega'_{D-d}, \omega'_{D-d+1} \ldots, \omega'_D)$. The max d-bounce irradiance represents the irradiance arriving at each patch after completing a path of length d following the given directions, and gathering and then reflecting the emission of the patches.

Limiting the solution to consider at most $D+1$ bounces, the solution of the rendering equation can be obtained as a $2D$-dimensional integral:

$$\mathbf{L}(\omega) = \left(\frac{1}{4\pi}\right)^D \int_{\Omega} \ldots \int_{\Omega} \left[\mathbf{L}^e(\omega) + 4\pi \cdot \mathbf{F}(\omega'_1, \omega) \cdot \mathbf{I}_D(\omega'_1, \ldots, \omega'_D)\right]\, d\omega'_D \ldots d\omega'_1. \qquad (6)$$

This high-dimensional integral can be estimated by Monte-Carlo quadrature.

3.2 Simple Monte-Carlo, or quasi-Monte Carlo integration

In order to evaluate formula (6), M random or quasi-random [10] walks should be generated (the difference is that in Monte-Carlo walks the directions are sampled randomly while in quasi-random walks they are sampled from a $2D$-dimensional low-discrepancy sequence, such as the $2D$-dimensional *Halton* or *Hammersley* sequence [16]). When the D-bounce irradiance is available, it is multiplied by the BRDF defined by the last direction ω_1 and the viewing direction ω to find a Monte-Carlo estimate of the radiance that is visible from the eye position. Note that this step makes the algorithm view-dependent. There are basically two different methods to calculate the estimate of the image. On the one hand, evaluating the BRDF once for each patch, a radiance value 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, based on the irradiance field, the surface normal and on the BRDF of the found point. In order to speed up this procedure, the surface visible at each pixel 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.

The final image is the average of these estimates. The complete algorithm — which requires just one variable for each patch i, the max d-bounce irradiance $\mathbf{I}[i]$ — is summarized in the following:

for $m = 1$ **to** M **do** *// samples of global walks*
  Generate the mth random or low-discrepancy point: $(\omega_1^{(m)}, \omega_2^{(m)}, \ldots, \omega_D^{(m)})$
  $\mathbf{I} = 0$
  for $d = 0$ **to** $D - 1$ **do** *// a random or quasi-random walk*
    $\mathbf{I} = \mathbf{A}(\omega'_{D-d}) \cdot \left(\mathbf{L}^e(\omega'_{D-d}) + 4\pi \cdot \mathbf{F}(\omega'_{D-d+1}, \omega'_{D-d}) \cdot \mathbf{I}\right)$
  endfor
  Calculate the image estimate from the irradiance $\mathbf{I}$
  Divide the estimate by M and add to the final image
endfor
Display Image

3.3 Combined and bi-directional walking techniques

The algorithm that has been derived directly from the quadrature formulae uses direction ω_1 to evaluate the contribution of 1-bounces, directions (ω_1, ω_2) for the 2-bounces, $(\omega_1, \omega_2, \omega_3)$ for the 3-bounces, etc. This is just a little fraction of the information that can be gathered during the complete walk. We could also use the samples of ω_1, ω_2, ω_3, etc. to calculate the 1-bounce contribution, (ω_1, ω_2), (ω_1, ω_3), ..., (ω_2, ω_3), etc. combinations of directions for 2-bounces, etc. This is obviously possible, since if the samples of $(\omega_1, \omega_2, \ldots \omega_D)$ are taken from a uniform sequence, then all combinations of its elements also form uniform sequences in lower dimensional spaces.

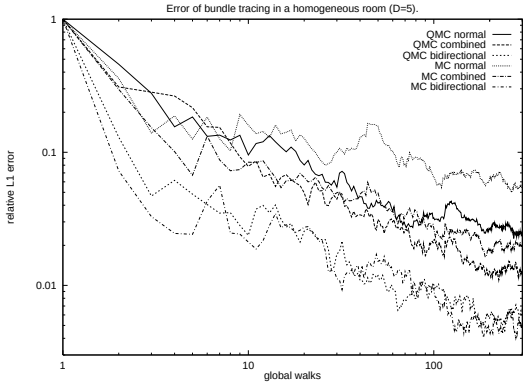

Fig. 3. Combined and bi-directional walking techniques versus normal walk

If all possible combinations are used, then each random walk generates $\binom{D}{d}$ samples for the d-bounces, which can be used to increase the accuracy of the method. Note that the increased accuracy of this "*combined*" method is for free in terms of additional computation. However, due to the dependence of the BRDF functions on two directions and due to the fact that different bounces will be estimated by different numbers of samples, the required storage per patch is increased to $D(D+1)/2$ variables. Since D is 5 to 8 in practical cases, this storage overhead is affordable. Furthermore, when the radiance is transferred to a direction, the required information to transfer the radiance to the

opposite direction is also available. Taking advantage of this, a single "*bi-directional*" walk will generate $\binom{D}{d} \cdot 2^d$ samples for the d-bounce transfer.

The additional samples of the "combined" and particularly the "bi-directional" walking techniques increase the accuracy as shown by figure 3. The test scene was the homogeneous Cornell-box where all surfaces have constant 0.5 diffuse reflectance and emission, which allowed to solve the rendering equation analytically [10] (the solution is $L = 1 - 2^{-(D+1)}$) to find a reference for the error analysis. Note that although quasi-Monte Carlo sampling is generally better, the improvement provided by the combined and bi-directional methods is less for the quasi-Monte Carlo walk than for the Monte-Carlo walk since the low-discrepancy points are so "well-designed" that mixing different sets of them does not improve the quadrature much further.

4 Metropolis solution of the directional integrals

The global illumination method proposed so far is particularly efficient if the lighting distribution in the scene does not exhibit high variations. For difficult lighting conditions importance sampling can help, that prefers those sequences of directions that transport significant radiance towards the eye. Since no a-priori information is available which these important directions are, some kind of adaptive technique must be used. In this section the application of Metropolis sampling is considered.

The Metropolis algorithm is a Monte-Carlo quadrature method that incorporates adaptive importance sampling by exploring the properties of the integrand automatically. Suppose that integral $I = \int_V f(\mathbf{z})\ d\mathbf{z}$ needs to be evaluated. Let us define importance $\mathcal{I}(\mathbf{z})$ that should be approximately proportional to f. Importance sampling requires the generation of samples $\{\mathbf{z}_1, \mathbf{z}_2, \ldots \mathbf{z}_M\}$ according to a probability density $p(\mathbf{z})$ — which is proportional to $\mathcal{I}(\mathbf{z}) = b \cdot p(\mathbf{z})$ — and using the following formula:

$$I = \int_V \frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})} \cdot \mathcal{I}(\mathbf{z})\ d\mathbf{z} = b \cdot \int_V \frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})} \cdot p(\mathbf{z})\ d\mathbf{z} = b \cdot E[\frac{f(\mathbf{z})}{\mathcal{I}(\mathbf{z})}] \approx \frac{b}{M} \cdot \sum_{i=1}^{M} \frac{f(\mathbf{z}_i)}{\mathcal{I}(\mathbf{z}_i)} \quad (7)$$

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 $p(\mathbf{z})$.

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

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

The generation of the next tentative sample is governed by a *tentative transition function* $T(\mathbf{x} \to \mathbf{y})$. In the algorithm we use a symmetric tentative transition function, that is $T(\mathbf{x} \to \mathbf{y}) = T(\mathbf{y} \to \mathbf{x})$. The transition probability of this Markovian process is:

$$P(\mathbf{x} \to \mathbf{y}) = \begin{cases} T(\mathbf{x} \to \mathbf{y}) & \text{if } \mathcal{I}(\mathbf{y}) > \mathcal{I}(\mathbf{x}), \\ T(\mathbf{x} \to \mathbf{y}) \cdot \mathcal{I}(\mathbf{y})/\mathcal{I}(\mathbf{x}) & \text{otherwise.} \end{cases} \quad (8)$$

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 (8), and then taking into account that the tentative transition function is symmetric, we can prove that the stationary probability distribution is really proportional to the importance:

$$\frac{p(\mathbf{x})}{p(\mathbf{y})} = \frac{P(\mathbf{y} \to \mathbf{x})}{P(\mathbf{x} \to \mathbf{y})} = \frac{T(\mathbf{y} \to \mathbf{x})}{T(\mathbf{x} \to \mathbf{y})} \cdot \frac{\mathcal{I}(\mathbf{x})}{\mathcal{I}(\mathbf{y})} = \frac{\mathcal{I}(\mathbf{x})}{\mathcal{I}(\mathbf{y})}. \tag{9}$$

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 (7).

4.1 Definition of the importance function

Let the importance function $\mathcal{I}$ be the sum of luminances of all pixels of the image, resulting from the walk. This importance function really concentrates on those walks that have a significant influence on the image. Using the luminance information is justified by the fact that the human eye is more sensitive to luminance variations than to color variations.

The Metropolis method achieves the sampling according to this probability by establishing a Markovian process on the space of global walks, whose limiting distribution is proportional to the selected importance function.

The Metropolis approximation of the radiance vector is:

$$\mathbf{L}(\omega) = (\frac{1}{4\pi})^D \int_{\Omega} \dots \int_{\Omega} \left[\mathbf{L}^e(\omega) + 4\pi \cdot \mathbf{F}(\omega'_1, \omega) \cdot \mathbf{I}_D(\omega'_1, \omega'_2, \dots, \omega'_D)\right] \, d\omega'_D \dots d\omega'_1 \approx$$

$$(\frac{b}{M}) \sum_{m=1}^{M} \frac{\mathbf{L}^e(\omega) + 4\pi \cdot \mathbf{F}(\omega'_1, \omega) \cdot \mathbf{I}_D(\omega_1^{(m)}, \omega_2^{(m)}, \dots, \omega_D^{(m)})}{\mathcal{I}(\omega_1^{(m)}, \omega_2^{(m)}, \dots, \omega_D^{(m)})}. \tag{10}$$

where M is the number of mutations and b is the integral of the importance function over the whole space.

4.2 Definition of the tentative transition function

The statespace of the Markovian process consists of D-dimensional vectors of directions that define the sequence of directions in the global walks. Thus the tentative transition function is allowed to modify one or more directions in these sequences.

4.3 Generating an initial distribution

The Metropolis method promises to generate samples with probabilities proportional to their importance in the stationary case. To ensure that the process is already in the stationary case from the beginning, initial samples are also selected according to the stationary distribution, i.e. proportionally to the importance function. Selecting samples with probabilities proportional to the importance can be approximated in the following way. A given number of seed points are found in the set of sequences of global directions. The importances of these seed points are evaluated, then, to simulate the distribution following this importance, the given number of initial points are selected randomly from these seed points using the discrete distribution determined by their importance.

4.4 Automatic exposure

Equation (10) also contains an unknown b constant that expresses the luminance of the image. The initial seed generation can also be used to determine this constant. Then at a given point of the algorithm the total luminance of the current image — that is the sum of the importances of the previous samples — is calculated and an effective scaling factor is found that maps this luminance to the expected one.

5 Calculation of the radiance transport in a single direction

In order to compute the irradiances of different bounces recursively by formula (6), the geometry matrix **A** must be determined for the actual direction and used to weight the radiance vector. Since the geometry matrix contains the relative areas of patches as visible from other patches of the scene, the determination of the geometry matrix requires the solution of a global visibility problem where the eye position visits each patch of the scene.

A possible solution is the application of the painter's algorithm that renders patches having sorted according to the given direction into an image buffer and computes the radiance transfer during rendering for each "pixel" [25].

In this section another method is proposed that traces back the visibility problem to a series of z-buffer steps to allow the utilization of the z-buffer hardware of workstations.

The proposed algorithm solves the discretized visibility problem placing a rasterized window perpendicularly to the given direction.

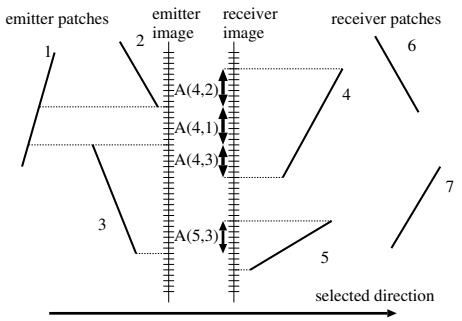

Fig. 4. Calculating the power transfer

The radiance is transferred by dynamically maintaining two groups of patches, an emitter group and a receiver group, in a way that no patch in the receiver group may hide a patch in the emitter group looking from the given direction.

Let the two classes of patches be rendered into two image buffers — called the emitter and receiver images, respectively — setting the color of patch j to j and letting the selected direction be the viewing direction for the receiver set and its inverse for the emitter set. Looking at figure 4, it is obvious that a pair of such images can be used to calculate the radiance transfer of all those patches which are fully visible in the receiver image. The two images must be scanned parallely and when i (that is the index of patch i) is found in the receiver image, the corresponding pixel in the emitter image is read and its value is used as a patch index to find the source radiance which is then added to the irradiance of patch i. Finally, the irradiance is scaled by P/A_i where P is the size

Fig. 5. A scene after the "first-shot"(left) and after 500 Metropolis walks (right)

of a pixel and A_i is the size of the receiver patch.

In order to find out which patches are fully visible in the receiver image, the number of pixels they cover is also computed during scanning and then compared to the size of their projected area. For those patches whose projected area is approximately equal to the total size of the covered pixels, we can assume that they are not hidden and their accumulated irradiances are valid, thus these patches can be removed from the receiver set and rendered into the emitter image to calculate the radiance transfer for other patches (this is the strategy to maintain the emitter and receiver sets automatically). This leads to an incremental algorithm that initially places all patches in the receiver set. Having calculated the receiver image by the z-buffer algorithm, the radiance transfer for the fully visible patches are evaluated, and then they are moved from the receiver set to the emitter set. The algorithm keeps doing this until no patch remains in the receiver set (cyclic overlapping would not allow the algorithm to stop, but this can be handled by a clipping as in the painter's algorithm [15]). The number of z-buffer steps required by the algorithm is quite small even for complex practical scenes [18]. Exploiting the built-in z-buffer hardware of advanced workstations, the computation can be fast.

6 Preprocessing of point lightsources

As other global radiosity methods, this method is efficient for large area lightsources but loses its advantages if the lightsources are small [19]. This problem can be solved by a "*first-shot*" that shoots the power of the point lightsources onto other surfaces, then removes them from the scene [2]. Since the surfaces can also be non-diffuse, the irradiance 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). The secondary, non-diffuse emission to a direction is computed from these irradiances.

7 Simulation results

Figure 5 shows a scene as rendered after the first shot and after 500 walks of length 5. The scene contains specular, metallic objects tessellated to 9602 patches, and is illuminated by both area (ceiling) and point (right-bottom corner) lightsources. A global radiance transfer took about 0.7 seconds on a Silicon Graphics O2 computer. Since the radiance information of a single patch is stored in 18 float variables (1 for the emission, 1 for the irradiance generated by the point lightsource, $D(D+1)/2 = 15$ for the irradiances and 1 for the accumulating radiance perceived from the eye), the extra memory used in addition to storing the scene is only 0.7 Mbyte.

The color plate shows a fractal terrain containing 14712 patches after 500 walks which provide an accuracy within 2 percents. A global radiance transfer took approximately 1.1 seconds and the radiance information required 1 Mbyte.

8 Conclusions

This paper presented a combined finite-element and random-walk algorithm to solve the rendering problem of complex scenes including also glossy surfaces. The basic idea of the method is to form bundles of parallel rays that can be traced efficiently, taking advantage of the z-buffer hardware. Unlike other random walk methods using importance sampling [7] [11] [28], this approach cannot emphasize the locally important directions, but handles a large number (1 million) parallel rays simultaneously instead, thus it is more efficient then those methods when the surfaces are not very specular.

The memory requirement is comparable to that of the diffuse radiosity algorithms although the new algorithm is also capable to handle non-diffuse reflections or refractions. Since global ray-bundle walks are computed independently, the algorithm is very well suited for parallelization.

In order to incorporate importance sampling, the Metropolis sampling technique was applied. However, for homogeneous scenes, we could not demonstrate significant noise reduction compared to quasi-Monte Carlo walks. This is due to the fact that the integrand of equation (4) is continuous and is of finite variation unlike the integrand of the original rendering equation, thus if its variation is modest then quasi-Monte quadrature is almost unbeatable. If the radiance distribution has high variation (difficult lighting conditions), then the Metropolis method becomes more and more superior. Future research should concentrate on the tuning of the Metroplis method in this application and on other adaptive importance sampling techniques.

References

1. 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.
2. F. Castro, R. Martinez, and M. Sbert. Quasi Monte-Carlo and extended first-shot improvements to the multi-path method. In *Proc. of SCCG'98*, pages 91–102, 1998.
3. 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.
4. 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.

5. R. Cook, T. Porter, and L. Carpenter. Distributed ray tracing. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, pages 137–145, 1984.
6. I. Deák. *Random Number Generators and Simulation*. Akadémia Kiadó, Budapest, 1989.
7. 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.
8. 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.
9. J. T. Kajiya. The rendering equation. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, pages 143–150, 1986.
10. 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.
11. E. Lafortune and Y. D. Willems. Bi-directional path-tracing. In *Compugraphics '93*, pages 145–153, Alvor, 1993.
12. 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.
13. 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.
14. L. Neumann. Monte Carlo radiosity. *Computing*, 55:23–42, 1995.
15. M. E. Newell, R. G. Newell, and T. L. Sancha. A new approach to the shaded picture problem. In *Proceedings of the ACM National Conference*, pages 443–450, 1972.
16. H. Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, Pennsilvania, 1992.
17. S. N. Pattanik and S. P. Mudur. Adjoint equations and random walks for illumination computation. *ACM Transactions on Graphics*, 14(1):77–102, 1995.
18. M. Sbert. *The Use of Global Directions to Compute Radiosity*. PhD thesis, Catalan Technical University, Barcelona, 1996.
19. M. Sbert, X. Pueyo, L. Neumann, and W. Purgathofer. Global multipath Monte Carlo algorithms for radiosity. *Visual Computer*, pages 47–61, 1996.
20. P. Shirley. Discrepancy as a quality measure for sampling distributions. In *Eurographics '91*, pages 183–194. Elsevier Science Publishers, 1991.
21. P. Shirley, C. Wang, and K. Zimmerman. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, 1996.
22. F. Sillion and C. Puech. A general two-pass method integrating specular and diffuse reflection. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, pages 335–344, 1989.
23. F. X. Sillion, J. R. Arvo, S. H. Westin, and D. P. Greenberg. A global illumination solution for general reflectance distributions. *Computer Graphics (SIGGRAPH '89 Proceedings)*, 25(4):187–198, 1991.
24. I. Sobol. *Die Monte-Carlo Methode*. Deutscher Verlag der Wissenschaften, 1991.
25. L. Szirmay-Kalos and T. Fóris. Sub-quadratic radiosity algorithms. In *Winter School of Computer Graphics '97*, pages 562–571, Plzen, Czech Republic, 1997.
26. E. Veach and L. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Rendering Techniques '94*, pages 147–162, 1994.
27. E. Veach and L. Guibas. Bidirectional estimators for light transport. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 419–428, 1995.
28. E. Veach and L. Guibas. Metropolis light transport. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 65–76, 1997.
29. J. R. Wallace, M. F. Cohen, and D. P. Greenberg. A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, pages 311–324, 1987.