Quasi-Monte Carlo Global Light Tracing with Infinite Number of Rays

László Szirmay-Kalos, Tibor Fóris, Werner Purgathofer
Department of Control Engineering and Information Technology, Technical University of Budapest,
Budapest, Műegyetem rkp. 11, H-1111, HUNGARY
szirmay@fsz.bme.hu

Abstract: The paper presents an efficient method to solve the general rendering equation, using a combined finite element and quasi-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 (that is non-diffuse) radiance. The direction dependent radiance function is then computed by quasi-random walk. Since quasi-Monte Carlo quadrature is applied here to an integrand of finite variation, this method can take advantage of the superior convergence of quasi-Monte Carlo integration.

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:

$$L(\vec{x},\omega) = L^e(\vec{x},\omega) + \int^{4\pi} L(h(\vec{x},-\omega'),\omega')\cdot\cos\theta'\cdot f_r(\vec{x},\omega',\omega)\, d\omega' \quad (1)$$

where $L(\vec{x},\omega)$ is the radiance of the surface in point $\vec{x}$ at direction ω, $h(\vec{x},\omega')$ 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 (figure 1).

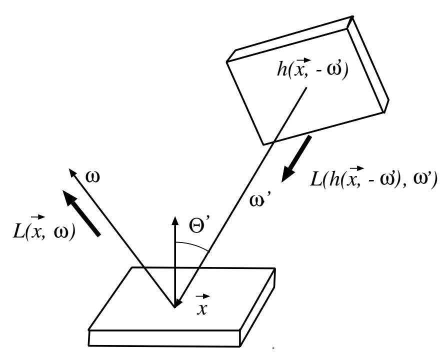

Figure 1: Geometry of the rendering equation

Generally two methods exist that can solve the rendering equation without simplifying the transport laws:

- *finite element methods* [CG85] which convert the integral equation into an approximating linear equation that can be solved by straightforward techniques,
- *random walk methods* [Sob91] that expand the rendering equation into a Neumann series, and evaluates its terms by multi-dimensional quadrature.

We encounter a serious difficulty when the finite element techniques are applied to the radiance equation. The function to be found is 4-variate and has high-frequency components, thus the number of required base functions, that approximate the solution as a linear combination, should be prohibitively high.

Random walk methods, on the other hand, require the solution of very high-dimensional integrals. The approximation error of integrals may be different from pixel to pixel, resulting in a noisy image. In order to reduce noise, the number of walks is increased and the image is generated as an average of the contributions of individual walks.

Due to the visibility term $h(\vec{x},-\omega')$ in the rendering equation, an estimation of dth bounce of the light requires the solution of d number of visibility problems. The most obvious way for this is ray-shooting which emanates a ray from point $\vec{x}$ at direction $-\omega'$ and finds the nearest intersection.

However, this approach may waste a lot of computation by ignoring all the intersection objects but the one closest to the eye. Thus it seems worth using a set of *global directions* [Sbe96][Neu95] 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. Other methods, 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.

Both random-walk and finite-element methods approximate high-dimensional integrals by finite sums. Since classical quadrature techniques based on uniform grids, such as trapezoidal rule, may have integration error of order $O(1/M^{1/s})$ where M is the number of dot points and s is the dimension of the domain, in order to guarantee a predefined error, the number of dot points should be in the order of $O(M^s)$, which is prohibitive for high-dimensional integrals. This phenomenon is called as the *dimensional explosion*.

The dimensional explosion can be avoided by *Monte-Carlo* [Sob91] or *Quasi-Monte Carlo* [Nie92] integration methods. Monte-Carlo methods trace back the estimation of an integral to the calculation of an expected value, which is estimated by averaging of random samples. Quasi-Monte Carlo techniques, on the other hand, use deterministic samples that are uniformly distributed in the integration domain. For the normalized, s-dimensional integration domain $[0,1]^s$, the quasi-Monte Carlo approximation is:

$$\int\limits_{[0,1]^s} f(\mathbf{x})\, d\mathbf{x} \approx \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{x}_i). \tag{2}$$

Sample points $\mathbf{x}_1, \ldots, \mathbf{x}_M$ should be selected to minimize the error of the integral quadrature.

If the integrand f has finite variation in the sense of Hardy and Krause, then the error of the quasi-Monte Carlo quadrature can be bounded as stated by the Koksma-Hlawka inequality:

$$|\int\limits_{[0,1]^s} f(\mathbf{x})\, d\mathbf{x} - \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{x}_i)| \leq \mathcal{V}_{\mathrm{HK}} \cdot D^*(\mathbf{x}_1, \ldots \mathbf{x}_M), \tag{3}$$

where $\mathcal{V}_{\mathrm{HK}}$ is the variation in the sense of Hardy and Krause and $D^*(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_M)$ is the star-discrepancy of the sample points [Nie92]. The star-discrepancy is defined by

$$D^*(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_M) = \sup_A |\frac{m(A)}{M} - V(A)| \tag{4}$$

where A is a s-dimensional subcube parallel to the coordinate axes and originating at the center, $V(A)$ is its volume, and $m(A)$ is the number of sample points inside this subcube.

For carefully selected sample points, called *low-discrepancy sequences* [Nie92], the discrepancy and consequently the error can be in the order of $O(\log^{s-1} M/M)$, which is much better than the $O(1/\sqrt{M})$ probabilistic error bound of Monte-Carlo methods. Moreover, quasi-Monte Carlo methods guarantee this accuracy in a deterministic way, unlike Monte-Carlo methods where the error bound is also probabilistic.

2 Previous work on non-diffuse radiosity

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* [ICG86] or *directional distributions* [Gla95], and to the application of *extended form factors* [SP89].

The first method calculates and stores radiance at each solid angle derived from a partitioned sphere. Extended form factors, on the other hand, represent both the direct-diffuse and the indirect-coherent light transfer of patches, and are usually evaluated by recursive ray-tracing.

Monte-Carlo random walk methods generate finite samples of the light propagation from the lightsources to the eye. Since a photon may bounce on surfaces several times until it reaches the eye, the propagation consists of several surface to surface interaction, called steps. The series of these steps composes the walk.

In computer graphics the first Monte-Carlo random walk algorithm — called *distributed ray-tracing* — was proposed by Cook et al. [CPC84], which spawned to a set of variations, including *path tracing* [Kaj86], *light-tracing* [DLW93], *Monte-Carlo radiosity* [Shi91][Neu95][PM95], and two-pass methods which combine radiosity and ray-tracing [WCG87].

Random-walks can be classified according to the direction they follow the light. A random-walk can be initiated from the lightsources and be traced in the direction of the light (shooting), or it can be started at the eye position and traced in the direction opposite to the light (gathering). Shooting type walks are also called *light walks*, while gathering type walks are called *eye walks*, emphasizing the starting point of the walk.

The problem of these methods is that the probability that a shooting path finds the eye is zero for a pinhole camera or very small if 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 random walk must be combined with a deterministic step that forces the walk to go to the eye or to find a lightsource. *Light tracing* connects each bounce position to the eye deterministically [DLW93]. *Bi-directional ray-tracing* methods start a shooting type and a gathering type walks and connect the bounce positions of the two walks. In a method proposed by Veach and Guibas [VG94] only the last positions of the two walks are connected. Lafortune [LW93], on the other hand, suggested that all positions visited during light-walks may be combined with the positions of eye-walks using a weighted sum.

The idea that visibility may be handled globally thus more efficiently was first appeared in [Neu95][Sbe96]. Different global radiosity methods incorporated different extensions of classical visibility algorithms (extension is required because here the eye position is not fixed). Neumann used an extended z-buffer in his transillumination method [Neu95], Sbert applied ray-shooting with global lines [Sbe96]. An $O(n \log n)$ version based on an extension of the painter's algorithm has been presented in [SKF97].

Quasi-Monte Carlo techniques have been first applied to solve the diffuse radiance equation by Keller [Kel95], where the integrand was generally discontinuous and therefore of infinite variation, thus the superiority of quasi-Monte Carlo method could not been theoretically justified. However, the numerical evidence showed that quasi-Monte Carlo methods can slightly be better than Monte-Carlo techniques. In [SKFNC97], we have provided a technique based on the transillumination method to reformulate the diffuse radiosity problem to make the integrand have finite variation.

3 Quasi-Monte Carlo global light tracing method

This section presents a new algorithm that decomposes the scene into a finite mesh. The light transfer between the patches is described by an integral equation consisting of surface and directional integrals. First, the surface integral is handled by finite element methods, then the resulting integral equation, which contains only directional integration, is solved by quasi-Monte Carlo walks.

3.1 Reformulation of the rendering equation using finite-element method

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 get:

$$L_i(\omega) \cdot A_i = L_i^e(\omega) \cdot A_i +$$

$$\int^{4\pi} \int_{A_i} L(h(\vec{x}, -\omega'), \omega') \cdot \cos\theta' \cdot \tilde{f}_i(\omega', \omega) \, d\vec{x} \, d\omega'. \quad (5)$$

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

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

$$\sum_{j=1}^{n} \tilde{f}_i(\omega', \omega) \cdot A(i, j, \omega') \cdot L_j(\omega'), \quad (6)$$

where $A(i, j, \omega')$ expresses the projected area of patch j that is visible from patch i. 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.

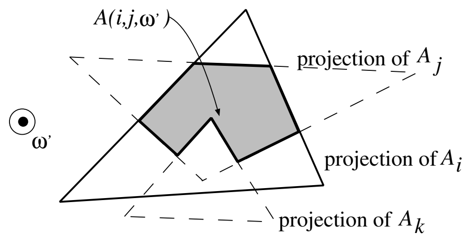

Figure 2: Interpretation of $A(i, j, \omega')$

Using this, the original radiance equation is:

$$L_i(\omega) \cdot A_i = L_i^e(\omega) \cdot A_i +$$

$$\int^{4\pi} \sum_{j=1}^{n} \tilde{f}_i(\omega', \omega) \cdot A(i, j, \omega') \cdot L_j(\omega') \, d\omega'. \quad (7)$$

Dividing both sides by A_i and converting the equation into matrix form, we obtain:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \int^{4\pi} \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega') \, d\omega', \quad (8)$$

where

$$\mathbf{L}(\omega) = [L_1(\omega), L_2(\omega), \ldots, L_n(\omega)],$$

is the vector of radiance values,

$$\mathbf{F}(\omega', \omega) = \langle \tilde{f}_1(\omega', \omega), \tilde{f}_2(\omega', \omega), \ldots, \tilde{f}_n(\omega', \omega) \rangle$$

is a diagonal matrix of BRDFs, and

$$\mathbf{A}(\omega') = \begin{bmatrix} 0 & \frac{A(1,2,\omega')}{A_1} & \ldots & \frac{A(1,n,\omega')}{A_1} \\ \frac{A(2,1,\omega')}{A_2} & 0 & \ldots & \frac{A(2,n,\omega')}{A_2} \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ \frac{A(n,1,\omega')}{A_n} & \frac{A(n,2,\omega')}{A_n} & \ldots & 0 \end{bmatrix}$$

is the matrix of visible projected areas.

Note that equation (8) 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 $A(i, j, \omega')/A_i$.

3.2 Quasi-random walk

In order to simplify the notations, the integral operator of the rendering equation is denoted by $\mathcal{T}$:

$$\int^{4\pi} \mathbf{F}(\omega', \omega) \cdot \mathbf{A}(\omega') \cdot \mathbf{L}(\omega')\, d\omega' = \mathcal{T}\mathbf{L}(\omega). \qquad (9)$$

Thus the short form of the rendering equation is:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \mathcal{T}\mathbf{L}(\omega). \qquad (10)$$

In equation (10) the unknown radiance function $\mathbf{L}(\omega)$ appears on both sides. Substituting the right side's $\mathbf{L}(\omega)$ by the complete right side, which is obviously $\mathbf{L}(\omega)$ according to the equation, we get:

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

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

$$\mathbf{L}(\omega) = \sum_{i=0}^{m} \mathcal{T}^i \mathbf{L}^e(\omega) + \mathcal{T}^{m+1}\mathbf{L}(\omega). \qquad (12)$$

If integral operator $\mathcal{T}$ is contractive — that is if $||\mathcal{T}\mathbf{L}(\omega)|| \leq \lambda \cdot ||\mathbf{L}(\omega)||$ and $\lambda < 1$ — then $\lim_{m\to\infty} \mathcal{T}^{m+1}\mathbf{L} = 0$, thus

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

The contractive property of $\mathcal{T}$ comes from the fact that a reflection or refraction always decreases the energy.

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.

Using the definition of integral operator $\mathcal{T}$, the full form of the Neumann series is:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) + \int^{4\pi} \mathbf{F}(\omega_1', \omega) \cdot \mathbf{A}(\omega_1') \cdot \mathbf{L}^e(\omega_1')\, d\omega_1' +$$
$$\int^{4\pi}\int^{4\pi} \mathbf{F}(\omega_1', \omega) \cdot \mathbf{A}(\omega_1') \cdot \mathbf{F}(\omega_2', \omega_1') \cdot \mathbf{A}(\omega_2') \cdot \mathbf{L}^e(\omega_2')\, d\omega_2' d\omega_1'$$
$$+ \ldots \qquad (14)$$

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 contractivity λ of operator $\mathcal{T}$ — that is the average reflectivity of the scene. Let us denote the maximum number of calculated bounces by D. The truncation of Neumann series introduces a bias in the estimation, which can be tolerated if D is high enough.

For D number of bounces the Neumann series can be considered as a single multi-dimensional integral:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) +$$
$$(\frac{1}{4\pi})^{D-1} \int^{4\pi}\int^{4\pi} \ldots \int^{4\pi} \mathbf{F}(\omega_1', \omega) \cdot \mathbf{A}(\omega_1') \cdot$$
$$\left(\mathbf{L}^e(\omega_1') + 4\pi \mathbf{F}(\omega_2', \omega_1') \cdot \mathbf{A}(\omega_2') \cdot (\ldots) \ldots\right) d\omega_D' \ldots d\omega_1'. \qquad (15)$$

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

$$\mathbf{I}_0(\omega_D') = \mathbf{A}(\omega_D') \cdot \mathbf{L}^e(\omega_D')$$
$$\mathbf{I}_d(\omega_{D-d}', \omega_{D-d+1}', \ldots, \omega_D') = \mathbf{A}(\omega_{D-d}') \cdot$$
$$\left(\mathbf{L}^e(\omega_{D-d}') + 4\pi \cdot \mathbf{F}(\omega_{D-d+1}', \omega_{D-d}') \cdot \mathbf{I}_{d-1}(\omega_{D-d-1}', \ldots \omega_D')\right). \qquad (16)$$

Using the max D-bounce irradiance, the rendering equation is:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) +$$
$$(\frac{1}{4\pi})^{D-1} \int^{4\pi}\int^{4\pi} \ldots \int^{4\pi} \mathbf{F}(\omega_1', \omega) \cdot \mathbf{I}_D(\omega_1', \ldots, \omega_D')\, d\omega_D' \ldots d\omega_1'. \qquad (17)$$

This high-dimensional integral can be estimated by quasi-Monte Carlo quadrature. Note that here the integrand is continuous and piece-wise continuously differentiable, which makes it of finite variation in the sense of Hardy and Krause. Thus we can really expect that the error of quadrature will be in the order of $O(\log^{2D-1} M/M)$ where M is the number of sample points.

3.3 Quasi-Monte Carlo integration

Since the low-discrepancy sequences are defined in the unit-cube, the directional sphere must be mapped into the unit cube, or alternatively, the points in the unit cube should be projected onto the directional hemisphere. In order to generate a uniform distribution on the surface of the directional sphere, the following transformation can be used [Sob91]:

$$\psi = 2\pi u, \qquad \theta = \arccos(1 - 2v). \qquad (18)$$

In this case

$$d\omega = det \begin{bmatrix} \frac{\partial\psi}{\partial u} & \frac{\partial\psi}{\partial v} \\ \frac{\partial\theta}{\partial u} & \frac{\partial\theta}{\partial v} \end{bmatrix} \cdot \sin\theta\, du\, dv = 4\pi \cdot du\, dv. \qquad (19)$$

thus the spatial distribution is uniform on the sphere if it was in the unit square.

The quasi-Monte Carlo approximation of the radiance vector is:

$$\mathbf{L}(\omega) = \mathbf{L}^e(\omega) +$$
$$(\frac{1}{4\pi})^{D-1} \int^{4\pi}\int^{4\pi} \ldots \int^{4\pi} \mathbf{F}(\omega_1', \omega) \cdot \mathbf{I}_1(\omega_1', \ldots, \omega_D')\, d\omega_D' \ldots d\omega_1' \approx$$
$$\mathbf{L}^e(\omega) + (\frac{4\pi}{M}) \sum_{i=1}^{M} \mathbf{F}(\omega_1^{(i)}, \omega) \cdot \mathbf{I}_D(\omega_1^{(i)}, \omega_2^{(i)}, \ldots, \omega_D^{(i)}). \qquad (20)$$

where M is the number of samples in the quadrature.

In order to evaluate this formula, M number of quasi-random walks should be generated, where the directions are sampled from a $2D$-dimensional low-discrepancy sequence, such as the $2D$-dimensional *Hammersley sequence* [Nie92]. The irradiance $\mathbf{I}$ can be computed recursively by equation (16) as the directions are processed from ω_D to ω_1. 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 the radiance of this patch that is visible from the eye position. Note that this step makes the algorithm view-dependent. The viewing direction is different from patch to patch if the eye is not at infinity. The complete algorithm, which requires two variables per patch, one for the max d-bounce irradiance $\mathbf{I}[p]$ and one for the accumulating radiance $\mathbf{L}[p]$, is summarized in the following:

for each patch p **do**
  Calculate viewing direction ω
  $\mathbf{L}[p] = \mathbf{L}^e(\omega)[p]$
endfor
for $i = 1$ **to** M **do**
  Generate the ith point: $(\omega_1^{(i)}, \omega_2^{(i)}, \ldots, \omega_D^{(i)})$
  $\mathbf{I} = 0$
  for $d = 0$ **to** $D - 1$ **do** // *quasi-random walk*
    $\mathbf{I} = \mathbf{A}(\omega'_{D-d})\cdot$
    $\big(\mathbf{L}^e(\omega'_{D-d}) + 4\pi \cdot \mathbf{F}(\omega'_{D-d+1}, \omega'_{D-d}) \cdot \mathbf{I}\big)$
  endfor
  for each patch p **do** // *accumulation*
    Calculate viewing direction ω
    $\mathbf{L}[p] += 4\pi/M \cdot \tilde{f}_p(\omega, \omega_1) \cdot \mathbf{I}[p]$
  endfor
endfor
Render Image

3.4 Improved walking techniques

The quasi-Monte Carlo algorithm that have been directly derived 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 information that can be gathered during the complete walk. Why do not we 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. combination of directions for 2-bounces, etc. as shown in figure 3. 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.

If all possible combinations are used, then each quasi-random walk generates $\binom{D}{k}$ number of samples for the k-bounces, which can be used to increase the accuracy of the method. Note that the increased accuracy is for free in terms of additional computation.

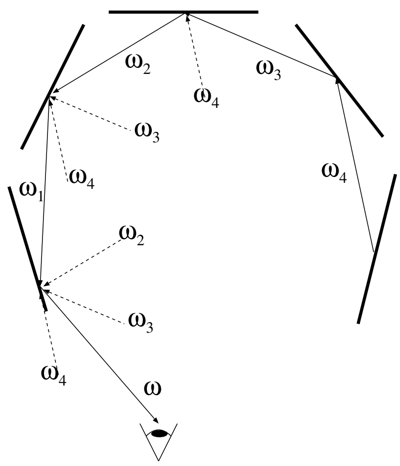

Figure 3: Improved walking technique versus quasi-random walk

However, due to the dependence of the BRDF functions onto two directions, the required storage per patch is increased to $D(D+1)/2 + 1$ number of variables. Since D is 5–8 in practical cases, this storage overhead is affordable.

This "improved walking technique" uses more samples to estimate the 1-bounce, 2-bounce, etc. contributions than the normal technique, thus we can expect a slight increase of the accuracy. However, if we use quasi-Monte Carlo integration, this increase is less than what could be observed for Monte-Carlo quadrature. The reason is that low-discrepancy sequences are so "well-designed" to be as uniform as possible, that mixing two different sequences cannot improve the discrepancy much further.

4 Calculation of visible areas

A crucial part of the method is the determination of areas $A(i, j, \omega')$ that express the projected area of patch j that is visible from patch i in direction ω'.

For a given i, it requires the solution of a visibility problem assuming the eye position in patch i and the viewing direction to be ω'. The visibility problem identifies those regions of the transillumination plane where patch j is visible. The areas of regions of different js should then be computed and summed to generate the needed function $A(i, j, \omega')$. Since $A(i, j, \omega')$ is needed for all i and j values in a fixed direction, this is a global visibility problem where the viewing direction is fixed but the eye position visits all patches. It would obviously be not very wise if we solved this problem handling different patches independently, but the patches should be processed parallely and the knowledge gained when the viewing position is at one patch must be reused for other patches as well. Theoretically, we could use either discrete or continuous visibility algorithms. Discrete algorithms that decompose the transillumination plane to small pixels are faster and the built-in rendering hardware and z-buffer can also be exploited, but in order to handle all patches simultaneously, the "window" of the algorithm should include all patches. The large window, however, should be decomposed into sufficiently small pixels to provide the required precision, which might result in unacceptable number of pixels. Continuous algorithms are free from these resolution problems, but are usually more difficult to implement and are slower.

Taking into account that the viewing direction determines which patch can be in front of another one, the first step of such continuous algorithm is the sorting of patches in the given viewing direction. Such a sorting is not obvious if the patches overlap in this direction, but this can be solved by clipping proposed in the painter's algorithms [NNS72].

In order to calculate $A(i, j, \omega')$ for patch i, only those j patches should be considered that are in front of patch i in the generated order. From the projection of these patches a planar graph is constructed where each region identifies a patch that is visible in it or specifies that no patch is visible here. Such a planar graph is usually called the *visibility map*. To compute $A(i, j, \omega')$ the regions of the visibility map are clipped onto the projection of A_i and the areas of internal regions identifying patch j as visible here are summed.

Note that this process is particularly suitable for incremental realization. When we step onto the next patch, the new visibility map is created by replacing those region parts that were inside the projection of A_i by the projection of A_i. The process of clipping of the patches onto each other is quite similar to the Weiler-Atherthon visibility algorithm [WA77].

The size of the visibility map is $O(n)$ in the average case but $O(n^2)$ in the worst-case. Thus the resulting algorithm that compares each patch with the actual visibility map will have $O(n^2)$ average case and $O(n^3)$ worst case time-complexity. Clearly this is not acceptable when complex scenes are processed. In order to reduce the complexity, we can use the wide selection of methods of computational geometry that usually apply spatial decomposition on the plane to reduce the number of unnecessary comparisons.

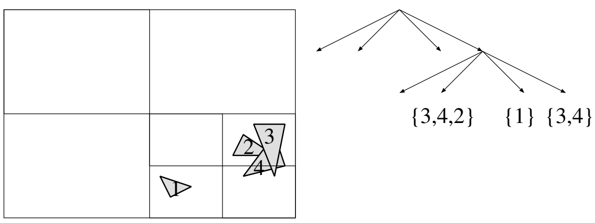

Figure 4: Quadtree-based spatial decomposition

The proposed decomposition subdivides rectangular areas into four sub-rectangles of the same size (figure 4). The corresponding data-structure is the quadtree where each leaf is associated with an ordered list of those regions that are inside the given rectangular area.

The quadtree that now represents the visibility map is constructed on the fly as the patches sorted in the viewing direction are processed. When a new patch is processed, first that portion of the visibility map is retrieved which is inside the given patch. This requires the identification of those leaves of the quadtree where the corresponding rectangle has non-empty intersection with the triangle. Then to prepare for the next patch, the inside portions are replaced by the current patch in these leaves. If during the processing the length of a list exceeds a given limit (say K), then the rectangle is decomposed into four sub-rectangles and its list is also subdivided into four not necessarily distinct lists. The subdivision stops if the size of the sub-rectangle is comparable to the size of the patches. Value K is set to the average number of patches that can be spearheaded by a ray parallel to the viewing direction in order to guarantee that the tree is balanced.

5 Simulation results

The presented algorithm has been implemented and tested. During simulations, the directions were generated from both quasi-Monte Carlo and from Monte-Carlo sequences. The simulations justified the superiority of quasi-Monte Carlo quadrature. The test scene was the Cornell-box which is a usual benchmark for random-walk methods [NNB97]. The low-discrepancy sequence has been generated by the incremental Halton generator proposed in [Kel96].

Figure 5: An image of the Cornell Box that has been generated by the proposed algorithm

6 Conclusions

This paper presented a light-tracing method which generates quasi-random walks and follows all light interactions that are parallel with the given directions.

This method can solve the non-diffuse radiosity problem — that is the general rendering equation — without increasing the number of free variables and the required storage as happens in earlier methods. Since in the finite-element phase the patches are assumed to have position independent radiance distribution, and in the random-walk phase the directional integrals are approximated by quasi-Monte Carlo quadrature, the method is accurate if the radiance function is quite smooth (perfect mirrors, for example, can produce artifacts).

Since the proposed method belongs to the category of global techniques, importance sampling based on the local BRDF cannot be exploited here. However, even for global directions it is often possible and also worthwhile to incorporate some form of importance sampling (for example in the test scenes global directions pointing down are more important since the lightsource is at the ceiling). Assigning importance to global directions is a non-trivial problem and is subject to further research. A promising possibility is to use Metropolis method [VG97] for a sequence of global directions.

The implemented method uses step-wise constant basis functions, which is not at all necessary. A further improvement, that we are currently working on, is the application of piece-wise linear basis function in the finite element phase. It has two benefits. On the one hand, linear basis functions can provide more accurate solution of the finite element phase. On the other hand, linear basis functions also make the integrand of the quasi-Monte Carlo integration smoother, which results in a faster convergence of the quadrature.

7 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.: 29ö4 and 32öu9).

References

[CG85] Michael Cohen and Donald Greenberg. The hemi-cube, a radiosity solution for complex environments. In *Proceedings of SIGGRAPH '85*, pages 31–40, 1985.

[CPC84] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. In *Proceedings of SIGGRAPH '84, Computer Graphics*, pages 137–145, 1984.

[DLW93] 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, Portugal, 1993.

[Gla95] A. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers, Inc., San Francisco, 1995.

[ICG86] David S. Immel, Michael F. Cohen, and Donald P. Greenberg. A radiosity method for non-diffuse environments. In *Proceedings of SIGGRAPH '86, Computer Graphics*, pages 133–142, 1986.

[Kaj86] James T. Kajiya. The rendering equation. In *Proceedings of SIGGRAPH '86, Computer Graphics*, pages 143–150, 1986.

[Kel95] 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.

[Kel96] A. Keller. The fast Calculation of Form Factors using Low Discrepancy Sequences. In *Proc. Spring Conference on Computer Graphics (SCCG '96)*, pages 195–204, Bratislava, Slovakia, 1996. Comenius University Press.

[LW93] E. Lafortune and Y.D. Willems. Bi-directional path-tracing. In *Compugraphics '93*, pages 145–153, Alvor, Portugal, 1993.

[Neu95] L. Neumann. Monte carlo radiosity. *Computing*, 55:23–42, 1995.

[Nie92] H. Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, Pennsilvania, 1992.

[NNB97] L. Neumann, A. Neumann, and P. Bekaert. Radiosity with well distributed ray sets. *Computer Graphics Forum (Eurographics'97)*, 16(3):261–270, 1997.

[NNS72] 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*, page 443, 1972.

[Pel97] M. Pellegrini. Monte carlo approximation of form factors with error bounded a priori. *Discrete and Computational Geometry*, 1997. (to appear).

[PM95] S.N. Pattanik and S.P. Mudur. Adjoint equations and random walks for illumination computation. *ACM Transactions on Graphics*, 14(1):77–102, 1995.

[Sbe96] M. Sbert. *The Use of Global Directions to Compute Radiosity*. PhD thesis, Catalan Technical University, Barcelona, 1996.

[Shi91] P. Shirley. Discrepancy as a Quality Measure for Sampling Distributions. In *Eurographics '91*, pages 183–194, Amsterdam, North-Holland, 1991. Elsevier Science Publishers.

[SKF97] L. Szirmay-Kalos and T. Fóris. Sub-quadratic radiosity algorithms. In *Winter School of Computer Graphics '97*, pages 562–571, Plzen, Czech Republic, 1997. University of West Bohemia.

[SKFNC97] 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.

[Sob91] I. Sobol. *Die Monte-Carlo Methode*. Deutscher Verlag der Wissenschaften, 1991. (in German).

[SP89] F. Sillion and C. Puech. A general two-pass method integrating specular and diffuse reflection. In *SIGGRAPH 89*, pages 335–344, 1989.

[VG95] Eric Veach and Leonidas Guibas. Bidirectional estimators for light transport. In *SIGGRAPH'95 Proceedings*, pages 419–428, 1995.

[VG97] E. Veach and L. Guibas. Metropolis light transport. *Computer Graphics Proceedings (SIGGRAPH'97)*, pages 65–76, 1997.

[WA77] Kevin Weiler and Peter Atherton. Hidden surface removal using polygon area sorting. In *Proceedings of SIGGRAPH '77, Computer Graphics*, pages 214–222, 1977.

[WCG87] John R. Wallace, Michael F. Cohen, and Donald P. Greenberg. A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. In *Proceedings of SIGGRAPH '87, Computer Graphics*, pages 311–324, 1987.