Lectures in Applied Mathematics
Volume **00**, 19xx

Quasi-Monte Carlo Methods in Computer Graphics: The Global Illumination Problem

Alexander Keller

Abstract. The main part of the global illumination problem of computer graphics is given by a Fredholm integral equation of the second kind, describing the light distribution in a closed environment. Calculating photorealistic images from that equation requires its kernel to be very complex and discontinuous. Due to this complexity Monte Carlo methods are an interesting tool for estimating a solution. In this article we investigate the application of deterministic quasi-Monte Carlo methods, as compared to pure Monte Carlo methods, for global illumination calculations in very complex environments and give numerical evidence for the superiority of the quasi-Monte Carlo methods.

1. Introduction

In photorealistic rendering in computer graphics a scene usually is given as a BREP (boundary representation). In addition to this geometry description, the surface properties such as roughness, texture and light emission are provided. The global illumination problem now consists in calculating an image from that description, taking into consideration all physical effects that emerge from this specification. The problem can be described by a second kind Fredholm integral equation. The kernel of that integral equation covers visibility, which is very expensive to compute and in addition makes the complexity of the integral equation strongly depend on the complexity of the BREP model of the scene. The visibility function has discontinuities, which are not aligned to any axis. So Monte Carlo methods seem to be a very useful approach for solving the equation. Instead of random sample points as used in the Monte Carlo method, we apply quasi-random sample points. These deterministic points lead to the quasi-Monte Carlo method, promising a faster convergence than the Monte Carlo rate. For a profound introduction to such points and quasi-Monte Carlo integration, see [**Nie92b**].

In the next section we will give a short introduction to quasi-Monte Carlo integration, followed by a brief description of the physical facts leading to the integral equation to solve. In section 4 we present an algorithm for the solution of the global illumination problem and finally come to the conclusion, that quasi-Monte Carlo

1991 *Mathematics Subject Classification.* Primary 65Y25, 65C05; Secondary 65R20.

0075-8485/00 $1.00 + $.25 per page

methods are superior to Monte Carlo methods for the global illumination problem, confirming the results of recent information based complexity theory issues [**TWW88**] [**Woź91**].

2. Quasi-Monte Carlo Integration

Monte Carlo integration is a powerful means whenever functions with unknown discontinuities have to be integrated. The *Monte Carlo method* estimates an integral by the mean value of N function values of the integrand f taken at the points $P_N = \{x_0, \dots, x_{N-1}\}$:

$$\int_{I^s} f(x)\, dx \approx \frac{1}{N} \sum_{i=0}^{N-1} f(x_i) \tag{2.1}$$

where $I^s = [0,1)^s$ is the s-dimensional unit cube and $P_N \subset I^s$. If the points P_N are chosen at random the error is expected to be

$$\mathbf{E} \left| \int_{I^s} f(x)\, dx - \frac{1}{N} \sum_{i=0}^{N-1} f(x_i) \right| \leq \frac{\sigma(f)}{\sqrt{N}} \tag{2.2}$$

where $\sigma(f)$ is the variance of the function f.

Since computers are deterministic machines, we are only able to generate pseudo-random numbers, approximating real random numbers. Their quality is checked by several statistical tests [**Nie92b**] [**Knu81**] and they are usually generated using the linear congruential method (for several other methods see [**Nie92a**]).

The most important property of the sampling point set P_N is uniform distribution, which also guarantees the convergence of the integration scheme (2.1) if f is Riemann-integrable [**Hla71**]. The deviation of the point set P_N from uniform distribution is measured by its *discrepancy*

$$D^*(P_N) := \sup_{J = \prod_{j=1}^s [0, a_j) \subset I^s} \left| \int_{I^s} \chi_J(x)\, dx - \frac{1}{N} \sum_{i=0}^{N-1} \chi_J(x_i) \right| .$$

So the discrepancy is the worst integration error for integrating the volume of all axis-aligned subcubes $J = \prod_{j=1}^s [0, a_j) \subset I^s$ by using P_N. A sequence of point sets P_N is called to be *uniformly distributed modulo* I^s if and only if

$$\lim_{N \to \infty} D^*(P_N) = 0 .$$

It can be proved, that the discrepancy of any point set P_N is bounded from below by

$$D^*(P_N) \geq B_s \frac{\log^{\frac{s-1}{2}} N}{N} \tag{2.3}$$

where $B_s > 0$ is a constant depending on the dimension s. When using random points in (2.1) we only have

$$D^*(P_N^{\text{random}}) \in \mathcal{O}\left(\sqrt{\frac{\log \log N}{N}} \right)$$

by the law of the iterated logarithm. This roughly is the rate (2.2) of the Monte Carlo method. But in fact there exist point sets and sequences, which acquire about the order of magnitude of (2.3) and therefore are called low discrepancy points. Using *low discrepancy point* as sample points in (2.1) is called *quasi-Monte*

Carlo integration. The construction of low discrepancy points is mostly based on the radical inverse function:

$$\Phi_b(i) := \sum_{j=0}^{\infty} a_j(i)\ b^{-j-1} \Leftrightarrow i = \sum_{j=0}^{\infty} a_j(i)\ b^j$$

For base $b = 2$ the radical inverse of some natural number $i \in I\!N_0$ simply is its binary representation mirrored at the decimal point. The most simple sequences are the Halton sequence and the Hammersley point set. The infinite Halton sequence in s dimensions is built by

$$x_i = (\Phi_{b_1}(i), \ldots, \Phi_{b_s}(i))$$

where $b_1, \ldots, b_s$ are the first s prime numbers, so as to reduce corellations between the components. Note that the number N of samples easily can be incremented by reusing all samples taken before. For the Halton points we have

$$D^*(P_N^{\text{Halton}}) < \frac{s}{N} + \frac{1}{N} \prod_{j=1}^{s} \left(\frac{b_j - 1}{2\ \log b_j} \log N + \frac{b_j + 1}{2} \right) .$$

The Hammersley point set is finite, that is the sample number N has to be fixed and adaptive sampling results in having to discard all samples taken so far:

$$x_i = (\frac{i}{N}, \Phi_{b_1}(i), \ldots, \Phi_{b_{s-1}}(i))$$

The disadvantage of being finite is paid off by the slightly superior rate of

$$D^*(P_N^{\text{Hammersley}}) < \frac{s}{N} + \frac{1}{N} \prod_{j=1}^{s-1} \left(\frac{b_j - 1}{2\ \log b_j} \log N + \frac{b_j + 1}{2} \right) .$$

Since those point sets are designed to minimize discrepancy, i.e. for integration or optimization purposes, they do not have random properties and therefore fail almost any statistical test. Using deterministic point sets for integration results in

THEOREM 2.1. *The Koksma-Hlawka inequality:*

$$\left| \int_{I^s} f(x)\ dx - \frac{1}{N} \sum_{i=0}^{N-1} f(x_i) \right| \leq V(f)\ D^*(P_N)$$

where $V(f)$ is the variation in the sense of Hardy and Krause.

So if the variation can be bounded by a constant, using low discrepancy point sets will result in a faster quadrature than using random numbers. But in computer graphics the integrands often are of the form

$$f(x) = g(x)\ \chi_{A_k}(h(x))\ p(x)$$

where $0 \leq g \in L_2$ is a bounded function, p is a probability density and χ_{A_k} is the characteristic function of the set $A_k \subset S$, telling whether the endpoint of the path $h(x)$ is in the set. Since χ_{A_k} usually is not aligned to any axis we have $V(f) = \infty$. In consequence theorem 2.1 is not applicable for integrals in computer graphics.

Instead we aim to make the integrand more smooth by using the importance sampling technique of Monte Carlo integration, which also applies to quasi-Monte Carlo integration for smooth functions [**SM94**]. In our case we are only able to

integrate the probability density function p. Let μ be the measure for the probability density p, then the integral can be rewritten:

$$\int_{I^s} g(x)\ \chi_{A_k}(h(x))\ p(x)\ dx = \int_{I^s} g(y)\ \chi_{A_k}(h(y))\ d\mu(y) \approx \frac{1}{N}\sum_{i=0}^{N-1} g(y_i)\ \chi_{A_k}(h(y_i))$$

The sample point cloud $C_N = \{y_0, \dots, y_{N-1}\}$ used for the Monte Carlo estimate approximates the density p. It is constructed out of the uniformly distributed point set P_N by means of the multi-dimensional inversion method (also known as Hlawka-Mück transformation [**HM72**]):

$$F_j(t_1, \dots, t_j) := \frac{\int_0^{t_j}\int_0^1 \cdots \int_0^1 p(t_1, \dots, t_{j-1}, \tau_j, \dots, \tau_s)\ d\tau_j \cdots d\tau_s}{\int_0^1\int_0^1 \cdots \int_0^1 p(t_1, \dots, t_{j-1}, \tau_j, \dots, \tau_s)\ d\tau_j \cdots d\tau_s}$$

From the equations

$$x^{(j)} = F_j(y^{(1)}, \dots, y^{(j)})$$

we successively determine

$$\begin{aligned} y^{(1)} \quad & \text{using} \quad x^{(1)} = F_1(y^{(1)})\ , \\ y^{(2)} \quad & \text{using} \quad x^{(2)} = F_2(y^{(1)}, y^{(2)}) \dots \end{aligned}$$

Then the inverse μ^{-1} is

$$\mu^{-1}(x) := (y^{(1)}, \dots, y^{(s)}) = y\ .$$

We now define the deviation of C_N from p, i.e. the discrepancy with respect to some density, by

$$D^*(p, C_N) := \sup_{J=\prod_{j=1}^s [0,a_j) \subset I^s} \left| \int_{I^s} \chi_J(x)\ p(x)\ dx - \frac{1}{N}\sum_{i=0}^{N-1} \chi_J(y_i) \right|\ .$$

Due to [**Wic74**] we have the following two theorems on the modeling of discrete densities:

THEOREM 2.2. *The discrepancy with respect to the probability density p is*

$$D^*(p, C_N) \le V(\mu^{-1})\ D^*(P_N)\ . \tag{2.4}$$

THEOREM 2.3. *If p is separable, i.e. $p(x) = \prod_{j=1}^s p^{(j)}(x^{(j)})$, we have*

$$D^*(p, C_N) = D^*(P_N)\ . \tag{2.5}$$

Since the variation of f is unbounded in computer graphics, we are not able to use theorem 2.1 for providing an error bound on the quadrature (2.1), although there exist deterministic point sets with lower discrepancy than random sample points. But theorems 2.4 and 2.5 apply for modeling densities. Using low discrepancy points then results in a better approximation than using random numbers. In consequence the quasi-Monte Carlo integration is faster than the Monte Carlo method, even in the setting of computer graphics.

3. The Global Illumination Problem

The global illumination problem consists of calculating an image of a three-dimensional scene description, taking into consideration all physical effects. The scene description is given by a boundary representation (BREP) of the objects to be illuminated. Usually this two-dimensional domain $S = \cup_{k=1}^{K} A_k$ is given by a disjoint union of surface primitives like for example triangles A_k. To guarantee energy preservation, the scene S must be closed. Photorealistic rendering requires realistic scene models. Such models usually consist of a very large number K of surface primitives (up to 10^6). In addition we are given the radiance emission $L_0 : S \times \Omega \to I\!R^+$ from a point on the surface into a direction of the hemisphere Ω in that point and the bidirectional reflectance distribution function (BRDF) $f_r : \Omega \times S \times \Omega \to I\!R^+$, depending on the incoming, outgoing direction, and the location, characterizing the reflectivity, roughness and color of the surfaces. For sake of simplicity we restrict ourselves to only the reflectance function in this paper and omit transluscent effects. The global illumination problem is given by two equations: The radiance equation describes the distribution of light in a scene whereas the pixel equation calculates the exposure of the radiances on an image.

Throughout this paper we use radiometric units, that is radiance L is given in electromagnetic power per unit projected area per unit solid angle. Usually L is given as a vector of some color primaries. In our simulation we selected the RGB - color system (red, green, blue). This system could easily be replaced by any other system without changing the algorithms described in the sequel. So $L = (r, g, b)$, where r, g and b are the intensities for the selected wavelengths of red, green and blue used for firing the electron guns of a CRT. For simplicity we use L as scalar in the algorithmic explanations. So whenever we write an equation using L, this is meant to be one equation for each component of the color base.

3.1. The radiance equation. The radiance equation describes the radiance distribution in the scene. It is given by a second kind Fredholm integral equation. The radiance L, which leaves from a point $x \in S$ on the surface of the scene in direction $\omega \in \Omega$, where Ω is the hemisphere in point x, is the sum of the source radiance L_0 and all reflected radiance (see figure 1 for the geometry):

$$\begin{aligned} L(x,\omega) &= L_0(x,\omega) + \int_\Omega L(h(x,\omega'),-\omega')\ f_r(-\omega',x,\omega)\ \cos\theta'\ d\omega' \\ &:= L_0 + T_{f_r} L \end{aligned} \tag{3.1}$$

Here $h(x,\omega') \in S$ is the first point that is hit when shooting a ray from x into direction ω'. This function accounts for visibility in the three-dimensional environment. Its calculation is the most expensive in the whole illumination process and is bounded from below by $\mathcal{O}(\log K)$. Minimizing the number of calls to h promises the fastest algorithms for the solution of (3.1). So the reflected radiance is the integral of the radiance of all points which can be seen through the hemisphere Ω in point x attenuated by the BRDF f_r and the projection term $\cos\theta'$, which puts the surface perpendicular to the ray (x,ω'). θ' is the azimuth angle between the surface normal in x and the direction ω'. Due to physical reasons we have $\|T_f\| < 1$, because any real scene is reflecting less than 100% of the radiance. Another important property used for the solution of the radiance equation is the Helmholtz principle:

$$f_r(-\omega',x,\omega) = f_r(-\omega,x,\omega')$$

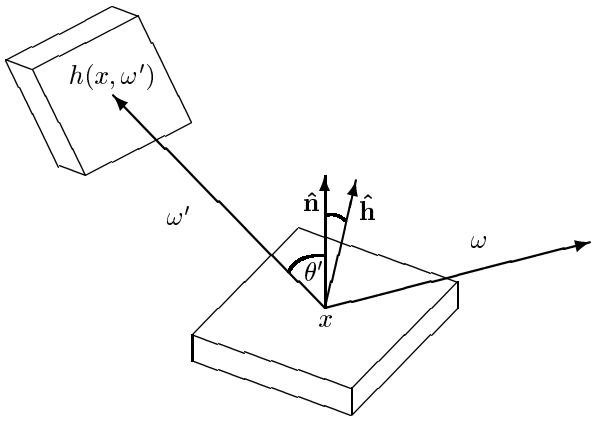

FIGURE 1. The geometry for the radiance equation.

This property allows to reverse all light paths.

3.2. The Pixel Equation. In order to calculate an image from the solution of the radiance equation (3.1), we have to simulate a camera lens. In our case we restrict ourselves to the most simple model of the pinhole camera. The radiance $\overline{L_P}$ of a pixel P of the rectangular image matrix then is the mean value integral of the radiance that is radiated through the area of that pixel into the eye:

$$\overline{L_P} = \frac{1}{|P|} \int_P L(x_{\text{Eye}}, x)\, dx \approx \sum_{i=0}^{OS-1} L(x_{\text{Eye}}, x_i) \tag{3.2}$$

Here $L(x_{\text{Eye}}, x)$ is the radiance, which emerges from the point seen from the eye-point x_{Eye} through the pixel position x into the direction of the viewpoint x_{Eye}. For a more realistic camera model we refer to [**KMH95**]. Choosing the set $P_{OS} = \{x_0, \ldots, x_{OS-1}\}$ is called antialiasing. For a survey on this subject and the application of quasi-Monte Carlo integration see [**HK94**].

4. Solving the Global Illumination Problem

The algorithm for the solution of the radiance equation (3.1) is split in two passes. The preprocessing step distributes the radiation starting from the light-sources by a random walk simulation. Then the rendering phase uses this information to reconstruct the radiances for the pixels in the image matrix.

4.1. Surface Properties. The surface properties are given by bidirectional reflection distribution functions (BRDF) of the kind

$$\begin{aligned} f_r(-\omega', x, \omega) &= f_d(x) + f_s(-\omega', x, \omega) \\ &= \frac{1}{\pi}\left(\rho_d(x) + \rho_s(x)\,\frac{1}{4\alpha}\,\frac{1}{\cos\theta'\cos\theta}\, e^{-\frac{\tan^2\delta}{\alpha^2}}\right) \end{aligned}$$

which were introduced by [**War92**]. $f_d = \frac{\rho_d(x)}{\pi}$ is the diffuse part of the BRDF, that is the fraction of radiance, which is reflected independently of incoming and outgoing direction. α is the root mean square of the surface slope. The f_s term accounts for specular reflection like mirrors ($\alpha = 0$) or glossy ($0 < \alpha < \infty$) objects.

The surface S is characterized by the reflectivities ρ_d and ρ_s (ρ_d and ρ_s, like L, are given in the color base (r,g,b) and account for the color of the reflection), and the roughness α. δ is the angle between the normal $\hat{\mathbf{n}}$ and the half vector $\hat{\mathbf{h}} = \frac{\omega+\omega'}{\|\omega+\omega'\|}$ of the directions ω and ω' in point x. θ is the angle between $\hat{\mathbf{n}}$ and ω. Although the structure of the BRDFs is simple, the model is sufficient for photorealistic image generation, fits physical experiments and is similar to theoretical derivations. Many more details like transparency or anisotopy can be added to the model but are omitted in this context.

A very important property of Ward's model is that the inversion method is applicable to f_r (see [**War92**]). So the process of photon scattering can be simulated in a very natural way. The reflected radiance hence is

$$\int_\Omega L(h(x,\omega'),-\omega')\ f_r(-\omega',x,\omega)\ \cos\theta'\ d\omega' = \int_\Omega L(h(x,\omega'),-\omega')\ dF_r(\omega')$$

In a simulation F_r then is modeled by choosing diffuse or specular scattering by the composition method and then scattering with the densities $f_d(x)\cos\theta'$ or $f_s(-\omega',x,\omega)\cos\theta'$, respectively, by using the inversion method. Note, that the distribution functions already include the projection term. Since we assumed all surfaces to be isotropic, the density f_r is separable in $(\theta',\phi') = \omega'$ and by theorem (2.5) using F_r for scattering does not change the discrepancy of the point set P_N used for modeling C_N. (Note, that F_r includes the projection term $\cos\theta'$.)

4.2. Algorithm. Each sample L of the pixel equation (3.2) now is calculated by the following decomposition of (3.1) :

$$\begin{array}{rcll} L & = & L_0 + T_{f_d+f_s} L & \\ & = & L_0 + T_{f_s} L + T_{f_d}(L_0 + T_{f_d+f_s} L) & \\ & = & L_0 & \text{Source radiation} \\ & + & T_{f_d} L_0 & \text{Direct illumination} \\ & + & T_{f_d} T_{f_d} L & \text{Indirect diffuse illumination} \\ & + & T_{f_s} L & \text{Specular effects} \\ & + & T_{f_d} T_{f_s} L & \text{Caustics} \end{array}$$

L_0 is given, $T_{f_d} L_0$ is evaluated by standard algorithms (see [**War91b**]). The remaining terms are evaluated using functionals of the form

$$\langle L, \Psi_D(\cdot_1,\cdot_2,\omega)\rangle = \int_S \int_\Omega L(y,\omega')\ \Psi_D(\omega', h(y,\omega'),\omega)\ \cos\theta'\ d\omega'\ dy \tag{4.1}$$

Depending on the "detector function" $\Psi_D : \Omega \times S \times \Omega \ \to I\!R^+$, those functionals return a part of radiance leaving in direction ω averaged over a domain D.

Indirect Diffuse Illumination. To calculate the indirect diffuse illumination, we proceed similar to [**Kel94**] by selecting

$$\Psi_{A_k}(\cdot_1,\cdot_2,\omega) = \frac{1}{|A_k|}\chi_{A_k}(\cdot_2)\ f_d(\cdot_2)$$

resulting in

$$\begin{array}{rcl} \overline{L_{A_k}} & := & \langle L, \Psi_{A_k}(\cdot_1,\cdot_2,\omega)\rangle \\ & = & \dfrac{1}{|A_k|}\ \displaystyle\int_S\int_\Omega L(y,\omega')\ \chi_{A_k}(h(y,\omega'))\ f_d(h(y,\omega'))\ \cos\theta'\ d\omega'\ dy\ . \end{array}$$

$\overline{L_{A_k}}$ is the mean radiance reflected by the diffuse part of f_r of the surface element A_k. Using

$$(T_{f_d} L)(x) \approx \overline{L}(x) := \sum_{k=1}^{K} \overline{L_{A_k}}\ \chi_{A_k}(x)$$

we approximate the indirect diffuse illumination by

$$(T_{f_d} T_{f_d} L)(x) \approx (T_{f_d} \overline{L})(x) = \int_\Omega \left(\sum_{k=1}^{K} \overline{L_{A_k}}\ \chi_{A_k}(h(x,\omega')) \right)\ f_d(x)\ \cos\theta'\ d\omega' \tag{4.2}$$

Specular Effects. The specular effects are treated using another functional:

$$\Psi_{B_r(x)}(\cdot_1, \cdot_2, \omega) = \frac{1}{\pi r^2}\ \chi_{B_r(x)}(\cdot_2)\ f_d(\cdot_2)$$

Here $B_r(x)$ is the ball around the point x with radius r and $\chi_{B_r(x)}(y)$ tells whether point y lies inside or outside this ball. πr^2 is approximately the area of the ball projected onto S. Hence

$$\begin{aligned} &(T_{f_d} L)(x) \\ &\quad\approx\quad \langle L, \Psi_{B_r(x)}(\cdot_1, \cdot_2, \omega) \rangle \\ &\quad=\quad \frac{1}{\pi r^2} \int_S \int_\Omega L(y,\omega')\ \chi_{B_r(x)}(h(y,\omega'))\ f_d(h(y,\omega'))\ \cos\theta'\ d\omega'\ dy \end{aligned}$$

approximates the mean radiant flux through B_r reflected by f_d in direction ω. The specular effects are treated by recursion:

$$T_{f_s} L = \begin{cases} T_{f_s}(L_0 + T_{f_r} L) & \text{max. level} \\ T_{f_s}(L_0 + T_{f_d} L + T_{f_s} L) & \text{else} \end{cases} \tag{4.3}$$

If the maximum level of recursion is reached we will estimate

$$\begin{aligned} &(T_{f_r} L)(x, \omega) \\ &\quad\approx\quad \langle L, \Psi'_{B_r(x)}(\cdot_1, \cdot_2, \omega) \rangle \\ &\quad=\quad \frac{1}{\pi r^2} \int_S \int_\Omega L(y,\omega')\ \chi_{B_r(x)}(h(y,\omega'))\ f_r(\omega', h(y,\omega'), \omega)\ \cos\theta'\ d\omega'\ dy \end{aligned}$$

by using the complete BRDF f_r.

Caustics. The effect of bright surface elements that are mirrored by specular onto diffuse surface elements is called caustic. Their detection and evaluation is a very hard problem in computer graphics. The caustics are approximated by:

$$\begin{aligned} &(T_{f_d} T_{f_s} L)(x) \\ &\quad\approx\quad \frac{1}{\pi r^2} \int_S \int_\Omega (T_{f_s} L)(y,\omega')\ \chi_{B_r(x)}(h(y,\omega'))\ f_d(h(y,\omega'))\ \cos\theta'\ d\omega'\ dy \end{aligned} \tag{4.4}$$

4.3. Implementation. The implementation is done in `C++`. The object oriented program system uses an extremely fast binary space partition (BSP) for the ray shooting $h(x,\omega)$. For specifying S, L_0 and f_r the material and geometry file format (MGF, see [**War95**]) is used.

Calculation of the Functionals. The preprocessing is a particle simulation of light, i.e. a random walk. Since $\|T_f\| < 1$ the Neumann series converges and can be cut off at a certain degree M, leaving a little underestimation since T_f and L are positive:

$$L = L_0 + T_f L = \sum_{i=0}^{\infty} T_f^i L_0 \approx \sum_{i=0}^{M} T_f^i L_0$$

We transform the functionals (4.1) into a sum of integrals on the unit cube, which then can be evaluated by random walk simulation using Monte Carlo methods:

$$\begin{aligned}
&\langle L, \Psi_D(\cdot_1, \cdot_2, \omega)\rangle \\
&\quad\approx\quad \langle \sum_{j=0}^{M} T_{f_r}^j\ L_0, \Psi_D(\cdot_1, \cdot_2, \omega)\rangle \\
&\quad=\quad \sum_{j=0}^{M} \langle T_{f_r}^j\ L_0, \Psi_D(\cdot_1, \cdot_2, \omega)\rangle \\
&\quad=\quad \sum_{j=0}^{M} \int_S \int_\Omega (T_{f_r}^j\ L_0)(y, \omega')\ \Psi_D(\omega', h(y, \omega'), \omega)\ \cos\theta'\ d\omega'\ dy \\
&\quad=\quad \sum_{j=0}^{M} \int_{\Omega^{j+1}} \int_{S_0} L_0(x_0, \omega_1)\ \Psi_D(\omega_{j+1}, x_{j+1}, \omega) \\
&\qquad\qquad \prod_{l=1}^{j} f_r(-\omega_l, x_l, \omega_{l+1})\ \prod_{l=1}^{j+1} \cos\theta_l\ dx_0\ d\omega_1 \cdots d\omega_{j+1} \\
&\quad=\quad \sum_{j=0}^{M} \int_{Q^{j+1}} \int_{S_0} L_0(x_0, \omega_1)\ \Psi_D(\omega_{j+1}, x_{j+1}, \omega) \\
&\qquad\qquad \prod_{l=1}^{j} f_r(-\omega_l, x_l, \omega_{l+1})\ \prod_{l=1}^{j+1} \frac{\sin 2\theta_l}{2}\ dx_0\ d\theta_1\ d\phi_1 \cdots d\theta_{j+1}\ d\phi_{j+1} \\
&\quad=\quad \sum_{j=0}^{M} |S_0|\ \pi^{2j+2} \int_{I^{2j+4}} L_0(x_0(u_0, u_1), \omega_1)\ \Psi_D(\omega_{j+1}, x_{j+1}, \omega) \\
&\qquad\qquad \prod_{l=1}^{j} f_r(-\omega_l, x_l, \omega_{l+1})\ \prod_{l=1}^{j+1} \frac{\sin \pi u_{2l}}{2}\ du_0 \cdots du_{2j+3} \qquad (4.5) \\
&\quad=\quad \sum_{j=0}^{M} |S_0|\ \pi^{j+1} \int_{I^{2j+4}} L_0(x_0(v_0, v_1), \omega_1)\ \Psi_D(\omega_{j+1}, x_{j+1}, \omega) \\
&\qquad\qquad \prod_{l=1}^{j} f_r(-\omega_l, x_l, \omega_{l+1})\ dv_0\ dv_1\ dF(v_2, v_3) \cdots dF(v_{2j+2}, v_{2j+3}) \qquad (4.6) \\
&\quad=\quad \sum_{j=0}^{M} |S_0|\ \pi^{j+1} \int_{I^{2j+4}} L_0(x_0(v_0, v_1), \omega_1)\ \Psi_D(\omega_{j+1}, x_{j+1}, \omega) \\
&\qquad\qquad dv_0\ dv_1\ dF(v_2, v_3)\ dF_r(v_4, v_5) \cdots dF_r(v_{2j+2}, v_{2j+3}) \qquad (4.7)
\end{aligned}$$

where

$$\begin{aligned} x_l &= h(x_{l-1}, \omega_l) \text{ for } l > 0 \ , \\ Q &= [0, \frac{\pi}{2}] \times [0, 2\pi] \text{ and} \\ \omega_l &= (\theta_l, \phi_l) = (\frac{\pi}{2} u_{2l}, 2\pi u_{2l+1}) \text{ in } x_{l-1} \ . \end{aligned}$$

S_0 is the surface of the lightsources where $L_0 > 0$. By (v_0, v_1) we access the point x_0 on S_0, using an area preserving mapping. For the Monte Carlo evaluation we now use a $(2M+4)$-dimensional low discrepancy sequence (u_i) for all decisions of the random walk. The random walk consists of N paths. For the generation of one path a particle with radiance $\frac{L_0}{N}$ is started in point x_0, modeled by $(u_{i,0}, u_{i,1})$, and traced in direction ω_1, modeled by $(u_{i,2}, u_{i,3})$. In $x_1 = h(x_0, \omega_1)$ its data (direction, position, radiance) is recorded. Then the particle is scattered and attenuated by using $(u_{i,4}, u_{i,5})$. This process is repeated for M reflections of the particle with the surface S. The particles are stored in an enhanced photon map similar to [**JC95**] and [**Jen95**]. A very compact representation is used, by compressing the particle information with the technique of [**War91a**]. The particles are arranged in a 3d-tree suited for range searching. For memory issues and fast traversal, this tree is balanced and stored without pointers in array representation. This approach avoids higher order FEM using spherical harmonics or similar attempts. Instead the storage usually needed for a fine tesselation and the basis coefficients is used for storing particles, which provide a more accurate information.

The formulation (4.5) is very inefficient, since many samples will be weighted by small values of the projection term. Importance sampling (or inversion) is used to avoid this effect in (4.6). The directions $(\frac{\pi}{2} v_{2l}, 2\pi v_{2l+1})$ are modeled with respect to the projection term:

$$dF(v_{2l}, v_{2l+1}) := d \sin^2 \frac{\pi}{2} v_{2l} \, dv_{2l+1}$$

In the experiments we will compare this kind of sampling to (4.7), where we also inverted the BRDF f_r. For the calculation of F_r, see [**War91b**]. This inversion prevents the samples to be strongly attenuated by specular BRDF, if the direction does not lie in the main reflection direction.

Indirect Diffuse Illumination. The $\overline{L_{A_k}}$ are calculated during the random walk. So whenever a particle hits S its radiance attenuated by the diffuse part of the BRDF f_r is added to the surface element in that point. The mean diffuse reflected radiance is used as a termination criterion. For two numbers N_1 and N_2 of iterated paths, we determine an error by

$$\triangle(N_1, N_2) := \frac{1}{\sum_{k=1}^{K} L_{0,k} \ |A_k|} \sqrt{\frac{\sum_{k=1}^{K} d(\overline{L_{A_k}(N_1)}, \overline{L_{A_k}(N_2)})^2 \ |A_k|}{\sum_{k=1}^{K} |A_k|}}$$

where the $\overline{L_{A_k}(N_i)}$ is the approximation by N_i paths and L_0 is the radiance of the different light sources. Further we select an interval $\triangle N$ for the measurements. The distance d is the Euclidean distance between the two color vectors $\overline{L_{A_k}(N_i)}$. The process is terminated if for a fixed T and the smallest $n \in I\!N$

$$\triangle((n+t) \triangle N, (n+t+1) \triangle N) < \epsilon \text{ for } 0 \le t < T \ .$$

Since the error is weighted by the size of the triangles, after termination the bigger areas A_k are integrated more exactly than the smaller areas. This makes sense,

because in the resampling step the bigger areas are hit more often than the smaller ones. The Monte Carlo evaluation of the indirect diffuse illumination (4.2) is

$$\begin{aligned}(T_{f_d}\overline{L})(x) &= \pi \int_\Omega \left(\sum_{k=1}^{K} \overline{L_{A_k}}\ \chi_{A_k}(h(x,\omega')) \right)\ f_d(x)\ dF(v_0,v_1) \\ &\approx \frac{\pi}{SCR}\ f_d(x) \sum_{i=0}^{SCR-1} \left(\sum_{k=1}^{K} \overline{L_{A_k}}\ \chi_{A_k}(h(x,\omega_i')) \right)\end{aligned}$$

The Monte Carlo calculation shoots SCR rays from the point x distributed over the hemisphere with respect to the $\cos\theta'$-distribution. The mean radiance of the element hit then contributes to the sum. The calculation is further enhanced by the *discontinuity buffer* (see [**Kel94**]). This techniques augments the sample number by using the samples of neighbouring pixels and so reduces the calls to $h(x,\omega)$ by roughly the factor 8.

Specular Effects. The specular illumination (4.3) is evaluated by taking one sample ω' in the reflection cone of the specular object generated by the density f_s. At the location hit in that direction we sample the source radiance and evaluate the diffuse reflected light by using the particles stored in the preprocessing step. For mathematical notation the P particles are accessible as $(L_p,\omega_p,x_p)_{p=1}^{P}$, where L_p is the radiance, ω_p is the incoming direction and x_p is the point where the particle hit the surface. The specular illumination is calculated by recursion:

$$(T_{f_s}L)(x,\omega) \approx \rho_s \left(L_0(y,-\omega') + \frac{f_d(y)}{\pi r^2} \sum_{p=1}^{P} L_p\ \chi_{B_r(y)}(x_p) + (T_{f_s}L)(y,-\omega') \right)$$

where $y = h(x,\omega')$ is the point hit when tracing the specular sample. If the maximum level of recursion is reached, we use the full BRDF f_r to estimate the total reflected radiance:

$$(T_{f_s}L)(x,\omega) \approx \rho_s \left(L_0(y,-\omega') + \frac{1}{\pi r^2} \sum_{p=1}^{P} L_p\ \chi_{B_r(y)}(x_p)\ f_r(\omega_p,y,-\omega') \right)$$

To evaluate these functionals, we sum over all particles, which fall into the ball $B_r(y)$. They are found using range searching.

Caustics. Particles which cause caustics are reflected by f_s and then hit a diffuse surface. Those particles' indices are recorded in the index set C in the preprocessing step. The evaluation of (4.4) now averages over the contribution of all caustic-particles in ball $B_r(x)$:

$$(T_{f_d}T_{f_s}L)(x) \approx \frac{f_d(x)}{\pi r^2} \sum_{p\in C} L_p\ \chi_{B_r(x)}(x_p)$$

This approximation is very crude and only for reasons of completeness. The major disadvantage of this method is, that small specular areas may not be hit in the preprocessing step, and that caustics from them are omitted although they may have very important contribution to the image.

Summary and Discussion. The algorithm and its implementation presented in this paper provide a full solution for the global illumination problem. The implementation uses a random walk preprocessing. Instead of a fine FEM-tesselation of the scene, particles of this step are stored in a very efficient structure. This structure enables the fast calculation of several kinds of radiance with very few calls

to the costly ray tracing function $h(x,\omega)$. As can be seen in the following section the number of paths, when using adaptive termination, is about the same order of magnitude for various scenes. So it would make sense to discard the $\overline{L_{A_k}}$ attached to the surface elements and to evaluate them by balls, too. Then the storage of the illumination information would become independent of the scene S and recursive and procedural scene modeling will be possible, allowing very complex scenes to be stored in very compact memory. The disadvantage of the algorithm, that caustics caused by small surface elements are likely to be missed, will be subject of further work. By omitting all contributions containing f_s, this algorithm also applies to the *radiosity problem* where $f_r \equiv f_d$. It is then very similar to [**Kel94**] (the difference lies in the treatment of f_d in the functional for the indirect diffuse illumination).

4.4. Numerical Evidence. The main interest of this work now is the evaluation of the functionals, especially the particle generation phase. For this part of the algorithm we want to compare the use of random numbers and quasi-random numbers, i.e. low discrepancy points for modeling particle densities with a large number of particles. All calculations were performed on an HP9000/735 99MHz workstation with 64MBytes of main memory in double precision.

For the experiments we used the `UNIX` random generator `drand48()` and the explicit inversive method [**Nie92a**] for the generation of pseudo-random numbers and the Halton and Faure [**SP87**] sequence for the low discrepancy sequences.

To first illustrate the power of using low discrepancy points (see also [**Kel95**]), we simplify the integral equation, so that an analytical solution exists for use as benchmark. Let $L_0 = \frac{1}{4}$ and $f_r = \frac{\rho_d}{\pi} = \frac{1}{2\pi}$ then we have

$$\begin{aligned} L &= L_0 + \int_\Omega \frac{\rho_d}{\pi} L \cos\theta(\omega)\, d\omega \\ &= L_0 + T_{f_r} L = \sum_{i=0}^{\infty} T_{f_r}^i L_0 \\ &= \sum_{i=0}^{\infty} \left(\frac{1}{2\pi}\right)^i \frac{1}{4} T^i = \frac{1}{4} \sum_{i=0}^{\infty} \frac{1}{2^i} = \frac{1}{2} \end{aligned}$$

where

$$T = \int_\Omega \cos\theta\, d\omega = \int_0^{2\pi} \int_0^{\frac{\pi}{2}} \cos\theta(\omega)\ \sin\theta\, d\theta\, d\phi = \pi$$

simply is the projection of the unit-hemisphere onto the plane. Then the solution is independent of the scene geometry ! For an illustration we used an empty 3-dimensional unit cube. In table 1 we see the number of paths needed with the parameters $\epsilon = 10^{-4}$, $2M+2 = 16$, $\Delta N = 1000$, and $T = 2$. The mean square deviation $\Delta_a(N)$ and the weighted mean square deviation $\Delta_{aw}(N)$ are defined as

$$\begin{aligned} \Delta_a(N) &:= \sqrt{\frac{\sum_{k=1}^K (\overline{L_{A_k}(N)} - \frac{1}{2})^2}{K}} \\ \Delta_{aw}(N) &:= \sqrt{\frac{\sum_{k=1}^K (\overline{L_{A_k}(N)} - \frac{1}{2})^2 |A_k|}{K \sum_{k=1}^K |A_k|}} \end{aligned}$$

where $\overline{L_{A_k}(N)}$ is the mean radiance after N paths of the random walk simulation. Obviously the low discrepancy points perform superior in this setting, i.e. they need less samples for the same accuracy than random sampling.

TABLE 1. Monte Carlo vs. quasi-Monte Carlo: Analytical Solution

	Monte Carlo		quasi-Monte Carlo	
	Congr.	Inversive	Halton	Faure
N	59000	64000	45000	50000
$\Delta_a(N)$	0.00599011	0.00632058	0.00536554	0.00543581
$\Delta_{aw}(N)$	0.000724511	0.00076198	0.000647638	0.000659806

In table 2 we show the comparison of Monte Carlo and quasi-Monte Carlo integration with and without importance sampling. It clearly can be seen, that the low discrepancy points lead to a faster convergence when using adaptive termination with the same values for ϵ, M and ΔN as above. In addition the importance sampling (the rows are marked by *inv.* for inversion) increases performance in specular scenes (the office and the cabin scene are mainly diffuse, and therefore no improvement is visible). The consequence is that modeling discrete densities, i.e. inverting as much of the integrand as possible, by low discrepancy points is superior to using random samples even for non-smooth integrands. The application of low discrepancy points even saves storage, since less particles are necessary to acquire the same level of accuracy when using random samples.

TABLE 2. Monte Carlo vs. quasi-Monte Carlo: Realistic Scenes

			N			
			Monte Carlo		quasi-Monte Carlo	
	Elements	Lights	Congr.	Inversive	Halton	Faure
office	6070	5	48000	51000	39000	41000
inv.			48000	51000	39000	38000
cabin	35346	8	30000	32000	25000	28000
inv.			30000	32000	25000	28000
doll	3604	89	80000	87000	86000	73000
inv.			60000	55000	48000	46000
soda	41420	5	70000	71000	65000	66000
inv.			54000	64000	55000	49000

To give an impression of the rendering results, we show some pictures of the scenes used for the measurements in figure 2.

FIGURE 2. Sample views of the scenes.

5. Conclusion

We presented an algorithm for the full solution of the global illumination problem in computer graphics. The design of the algorithm was guided by means of variance reduction techniques from the Monte Carlo method. By experimental evidence we showed, that those techniques, which also can be applied for variation reduction in quasi-Monte Carlo methods, can be applied even if the integrands have infinite variation. The experiments also showed, that the quasi-Monte Carlo methods are superior to Monte Carlo methods, i.e. they converge faster.

References

[HK94] S. Heinrich and A. Keller, *Quasi-Monte Carlo methods in computer graphics, Part I: The QMC-Buffer*, 242/94, University of Kaiserslautern, 1994.

[Hla71] E. Hlawka, *Discrepancy and Riemann Integration*, Studies in Pure Mathematics (New York) (L. Mirsky, ed.), Academic Press, New York, 1971, pp. 121–129.

[HM72] E. Hlawka and R. Mück, *Über eine Transformation von gleichverteilten Folgen II*, Computing (1972), no. 9, 127–138.

[JC95] H. Jensen and N. Christensen, *Photon maps in birdirectional monte carlo ray tracing of complex objects*, Computer Graphics **19** (1995), no. 2, 215–224.

[Jen95] H. Jensen, *Importance driven path tracing using the photon map*, Rendering Techniques '95, 1995, pp. 326–335.

[Kel94] A. Keller, *A quasi-Monte Carlo algorithm for the global illumination problem in the radiosity setting*, 260/94, University of Kaiserslautern, 1994.

[Kel95] A. Keller, *Quasi-Monte Carlo Methods in Computer Graphics*, ZAMM (1995), 0.

[KMH95] C. Kolb, D. Mitchell, and P. Hanrahan, *A realistic camera model for computer graphics*, Computer Graphics Proceedings, August 1995, pp. 317–324.

[Knu81] D. Knuth, *The Art of Computer Programming Vol. 2: Seminumerical Algorithms*, Addison Wesley, 1981.

[Nie92a] H. Niederreiter, *New methods for pseudorandom number and pseudorandom vector generation*, Winter Simulation Conference (Arlington, VA, 1992) (Piscataway, NJ), IEEE Press, 1992, pp. 264–269.

[Nie92b] H. Niederreiter, *Random number generation and quasi-Monte Carlo methods*, SIAM, Pennsylvania, 1992.

[SM94] J. Spanier and E. Maize, *Quasi-random methods for estimating integrals using relatively small samples*, SIAM Review **36** (1994), no. 1, 18–44.

[SP87] P. Sarkar and M. Prasad, *A comparative study of pseudo and quasi random sequences for the solution of integral equations*, J. Comp. Physics (1987), no. 68, 66–88.

[TWW88] J. Traub, G. Wasilkowski, and H. Woźniakowski, *Information-Based Complexity*, Academic Press, 1988.

[War91a] G. Ward, *Real pixels*, Graphics Gems II (J. Arvo, ed.), Academic Press, 1991, pp. 80–83.

[War91b] G. J. Ward, *Adaptive shadow testing for ray tracing*, 2nd Eurographics Workshop on Rendering (Barcelona, Spain), 1991.

[War92] G. J. Ward, *Measuring and Modeling Anisotropic Reflection*, Computer Graphics, July 1992, pp. 265 – 272.

[War95] G. Ward, Realistic Input for Realistic Images, ch. The Materials and Geometry Format, ACM SIGGRAPH Course Notes, 1995.

[Wic74] J. Wick, *Zur Anwendung der Approximation durch endliche Punktmengen auf die Lösung von Integro-Differentialgleichungen*, Jül-1124-MA, Zentralinstitut für Angewandte Mathematik, Kernforschungsanlage Jülich, Oktober 1974.

[Woź91] H. Woźniakowski, *Average case complexity of multivariate integration*, Bull. Amer. Math. Soc. **24** (1991), 185–194.

Department of Computer Science, Kaiserslautern University, Postfach 3049, D-67653 Kaiserslautern, Germany

E-mail address: keller@informatik.uni-kl.de, http://www.uni-kl.de/AG-Heinrich