Interner Bericht

Quasi-Monte Carlo Methods in Computer Graphics, Part I: The QMC-Buffer

Stefan Heinrich
Alexander Keller

Fachbereich Informatik

Universität Kaiserslautern · Postfach 3049 · D-67653 Kaiserslautern

Quasi-Monte Carlo Methods in Computer Graphics, Part I: The QMC-Buffer

Stefan Heinrich
Alexander Keller

Fachbereich Informatik
AG Numerische Algorithmen
Universität Kaiserslautern
Postfach 3049, D-67653 Kaiserslautern
e-mail: heinrich@informatik.uni-kl.de
keller@informatik.uni-kl.de

242/94

April 11, 1994

Abstract

Monte Carlo integration is often used for antialiasing in rendering processes. Due to low sampling rates only expected error estimates can be stated, and the variance can be high. In this article quasi-Monte Carlo methods are presented, achieving a guaranteed upper error bound and a convergence rate essentially as fast as usual Monte Carlo.

1 Introduction

Monte Carlo methods are by now widely used in the process of rendering photorealistic images in computer graphics. As in many other fields, the essence of application of Monte Carlo methods is usually the evaluation of certain multidimensional integrals.

Intensive investigations in the theory of multivariate integration [NIE78], [NIE92] and recently also in information-based complexity theory [TRA88], [WOZ91] have brought up an alternative approach: Quasi-Monte Carlo methods. These are deterministic quadrature rules specifically fitted to multidimensional integration. Requiring only a mild regularity of the integrand, they provide a considerably smaller error than standard Monte Carlo, and moreover, this error estimate is deterministic, i.e. guaranteed. Quasi-Monte Carlo methods replace random numbers by elements of low discrepancy sequences, which are sequences of points which approximate the equidistribution in a multidimensional cube in an optimal way.

So far low discrepancy sequences and quasi-Monte Carlo methods have not yet been applied in computer graphics. We think, however, that they might be quite useful in this field, too. In the present paper we study the question of pixel oversampling in the process of rendering realistic images from this point of view.

For details and bibliography on quasi-Monte Carlo methods we refer to [NIE92]. Notation, facts and methods of rendering can be found in [FOL90] and [WAT93]. Antialiasing by oversampling is treated in [COO84], [DIP85], [LEE85], [PAI89], [MIT87] and [MIT91].

Since this paper addresses both specialists in computer graphics and multivariate integration, we include certain explanations of basic notions and facts from either fields.

2 Oversampling

An image of a (real) scene is obtained by putting a rectangular screen between the eye (point) of the spectator and the scene. Through each point of the screen a ray is sent from the eye into the scene. The light radiation from this point of the scene towards the eye determines the color and brightness of the point on the screen.

For simplicity we consider only one selected wavelength, i.e. monochromatic light (the full image is then obtained by superposition). So to each point $x = (u, v)$ of the screen there corresponds a radiance $L(u, v)$. In practice, the screen has to be divided into pixels P_i and the computer image is supposed to show the average

radiance $\bar{L}_i$ over P_i,

$$\bar{L}_i = \frac{1}{|P_i|} \int_{P_i} L(x)\, dx \tag{1}$$

where $|P_i|$ denotes the area of P_i. The simple choice of approximating this integral by one point, e.g. the midpoint of P_i, can lead to large deviations from (1) and, in particular, to aliasing ([WAT93], [COO84]). On the other hand, we cannot spend too many points on one pixel since the screen usually consists of around $2.5 \cdot 10^5$ to 10^6 pixels. Moreover, in certain situations (as e.g. global illumination of complex scenes) it is very time consuming to obtain a value of the integrand $L(u, v)$. (In fact, the values of L can be high dimensional integrals themselves. The quasi-Monte Carlo approach for their computation is treated in [HEI94].) This sets the task of finding suitable integration rules for (1) using as little samples as possible (say from 3 to 50, to indicate the possible range).

The integrand is usually only piecewise smooth, and we do not know the lines of jump discontinuities in advance. So higher order numerical quadrature does not work well. Therefore various forms of Monte Carlo sampling have been proposed for the computation of (1). We discuss some of them in section 6. Let us also mention that using the same sampling pattern for all pixels is desirable, because it leads to highly efficient implementations, since buffer techniques or even hardware support can be applied (e.g. the accumulation-buffer [HAE90], Z-buffer or item-buffer). In the present paper we confine our analysis to this case. It should be mentioned, however, that patterns varying from pixel to pixel promise better antialiasing. We shall discuss low discrepancy approaches to this situation in a future work.

Let the pixel P under consideration be parametrized in such a way that $P = B = [0, 1]^2$. Since a pixel usually covers a small sector of the scene, we can assume a simple geometry of the discontinuities of $L(u, v)$. In fact, scenes in photorealistic rendering are often built from triangular elements. So we shall assume that P can be decomposed into the sum of (a few) subsets of very simple geometry (e.g. triangles)

$$P = \cup_{k=1}^{K} C_k \ .$$

in such a way, that the restriction of L to C_k is sufficiently smooth (precise requirements will be given below), or even of a given form, as e.g. linear. Hence we are faced with the problem of optimally integrating piecewise smooth functions with varying and a priory unknown domains of smoothness. By the physical meaning, $L(u, v) \geq 0$, and it is reasonable to assume a uniform bound $L(u, v) \leq L_0$, since the technical limits of a screen restrict the maximally representable intensity anyway. Finally, the number of basic sets is assumed to be bounded by a (small) constant, $K \leq K_0$, that is we do not allow, in particular, fractal surfaces or infinite geometries.

3 Quasi-Monte Carlo Integration

Let $B = [0,1]^s$ be the s-dimensional unit cube and f be a square integrable function. The Monte Carlo method approximates the integral of f by

$$\int_B f(x)\,dx \approx \frac{1}{N}\sum_{i=0}^{N-1} f(x_i)\ ,$$

where $\{x_i \in B \mid 0 \le i < N\}$ is a set of random points. For large N, the central limit theorem states that the deviation of the sum from the integral approaches a normal distribution, and a convergence rate of $O\left(\frac{1}{\sqrt{N}}\right)$ sets in with high probability [SOB91], [NIE92]. The low number of samples, however, violates this assumption and only the mean square error can be affirmed:

$$\sqrt{I\!E\left(\int_B f(x)\,dx - \frac{1}{N}\sum_{i=0}^{N-1} f(x_i)\right)^2} = \frac{\sigma}{\sqrt{N}}\ .$$

with σ^2 being the variance of f.

To overcome the probabilistic error of pure Monte Carlo, the random sampling pattern is replaced now by a deterministic pattern. The calculation of the mean pixel color stays the same and the method is then called quasi-Monte Carlo integration due to the use of quasi-random numbers. Quasi-random numbers are deterministic point sets, which approximate the uniform distribution as much as possible. This can be expressed quantitatively by the notion of discrepancy.

The discrepancy is a measure of the uniform distribution of a given (deterministic) point set $P_N = \{x_0, \ldots, x_{N-1}\}$. It is defined with respect to a family of subsets of B. So let $\mathcal{A}$ be a non-empty set of Lebesgue-measurable subsets of B. We define the discrepancy of P_N with respect to $\mathcal{A}$ as

$$D(P_N, \mathcal{A}) = \sup_{A \in \mathcal{A}} \left| \lambda_s(A) - \frac{m(A, P_N)}{N} \right|\ , \tag{2}$$

where λ_s is the Lebesgue measure on B, and

$$m(A, P_N) = \sum_{i=0}^{N-1} \chi_A(x_i) \tag{3}$$

is the number of elements of P_N which fall into A. Inserting (3) into (2) yields another interpretation: $D(P_N, \mathcal{A})$ is the maximal error of the quadrature formula

defined by P_N (with equal weights $\frac{1}{N}$) when integrating characteristic functions of elements from $\mathcal{A}$.

The classical or extreme discrepancy $D(P_N)$ is obtained for

$$\mathcal{A} = \left\{ \prod_{i=1}^{s} [a_i, b_i] \mid 0 \le a_i < b_i \le 1, i = 1, \ldots, s \right\} .$$

When we consider the family of all such parallelepipeds with $a_1 = a_2 = \cdots = a_s = 0$, we get the star discrepancy $D^*(P_N)$. The isotropic discrepancy $J(P_N)$ arises when $\mathcal{A}$ is taken to be the family of all convex subsets of B (see [NIE92] for these definitions). For the purposes of analyzing pixel oversampling ($s = 2$) it seems appropriate to consider further classes $\mathcal{A}$ formed of sets which reflect the basic geometry occurring most frequently in pixels: The class $\mathcal{T}$, where $A \in \mathcal{T} \Leftrightarrow A$ is a triangle contained in B, or the class $\mathcal{E}$ of edges (lines through the pixel), where $A \in \mathcal{E} \Leftrightarrow A$ is the intersection of B with a half-plane. In the next section we analyze the role of discrepancy for pixel integration.

4 Deterministic Error Bounds

Using the notions of discrepancy one can estimate the deterministic error of the quasi-Monte Carlo quadrature, given by a point set P_N. For the sake of simplicity we consider only the case of smooth integrands and $s = 2$. Let $f : [0,1]^2 \to \mathbb{R}$ be a function with continuous mixed derivative $\frac{\partial^2 f(u,v)}{\partial u \, \partial v}$. The variation of f in the sense of Hardy and Krause is defined as

$$V(f) = V^{(1,1)}(f) + V^{(1,0)}(f) + V^{(0,1)}(f)$$

where

$$\begin{aligned} V^{(1,1)}(f) &= \int_0^1 \int_0^1 \left| \frac{\partial^2 f(u,v)}{\partial u \, \partial v} \right| du \, dv \\ V^{(1,0)}(f) &= \int_0^1 \left| \frac{\partial f(u,1)}{\partial u} \right| du \\ V^{(0,1)}(f) &= \int_0^1 \left| \frac{\partial f(1,v)}{\partial v} \right| dv \, . \end{aligned}$$

(For the general definitions we refer to [NIE92].) The classical Koksma-Hlawka inequality for integration on $[0,1]^2$ states that for any set $P_N = \{x_0, \ldots, x_{N-1}\} \subset [0,1]^2$

$$\left| \int_{[0,1]^2} f(x) \, dx - \frac{1}{N} \sum_{i=0}^{N-1} f(x_i) \right| \le V(f) \cdot D^*(P_N) \qquad (4)$$

(see [NIE92]). Due to the discontinuities of the radiance function $L(x)$ we cannot apply (4) directly. Instead we have to integrate over subregions of the square, and we need an analogue of (4) for this situation. Such a result was obtained by Zaremba [ZAR70] (see also Niederreiter [NIE72] and de Clerk [DCL81]). We shall restate Zaremba's result in a form suited for our purposes. Let $\mathcal{A}$ be a non-empty class of Lebesgue measurable subsets of the square. Denote by

$$\begin{aligned}
\mathcal{A}^{(1,1)} &= \{A \cap [0,a] \times [0,b] \mid A \in \mathcal{A}, 0 \le a,b \le 1\} \\
\mathcal{A}^{(1,0)} &= \{A \cap [0,a] \times [0,1] \mid A \in \mathcal{A}, 0 \le a \le 1\} \\
\mathcal{A}^{(0,1)} &= \{A \cap [0,1] \times [0,b] \mid A \in \mathcal{A}, 0 \le b \le 1\} .
\end{aligned}$$

The following is basically Zaremba's proposition 2.1 [ZAR70]:

Proposition 4.1 *For all $P_N = \{x_0, \ldots, x_{N-1}\} \subset [0,1]^2$, all functions f with continuous mixed derivative $\frac{\partial^2 f}{\partial u\, \partial v}$ on $[0,1]^2$ and all $A \in \mathcal{A}$*

$$\begin{aligned}
&\left| \int_A f(x)\, dx - \frac{1}{N} \sum_{\substack{i=0 \\ x_i \in A}}^{N-1} f(x_i) \right| \\
&\le f(1,1) \cdot D(P_N, \mathcal{A}) + V^{(1,1)}(f) \cdot D(P_N, \mathcal{A}^{(1,1)}) \\
&\quad + V^{(1,0)}(f) \cdot D(P_N, \mathcal{A}^{(1,0)}) + V^{(0,1)}(f) \cdot D(P_N, \mathcal{A}^{(0,1)}) .
\end{aligned}$$

For the sake of completeness we indicate the idea of the proof:

Proof: Integration by parts gives

$$f(u,v) = f^{(1,1)}(u,v) - f^{(1,0)}(u,v) - f^{(0,1)}(u,v) + f(1,1) .$$

where

$$\begin{aligned}
f^{(1,1)}(u,v) &= \int_0^1 \int_0^1 \chi_{[0,a]}(u) \chi_{[0,b]}(v) \frac{\partial^2 f(a,b)}{\partial a\, \partial b} da\, db \\
f^{(1,0)}(u,v) &= \int_0^1 \chi_{[0,a]}(u) \frac{\partial f(a,1)}{\partial a} da \\
f^{(0,1)}(u,v) &= \int_0^1 \chi_{[0,b]}(v) \frac{\partial f(1,b)}{\partial b} db .
\end{aligned}$$

We shall only mention how to handle $f^{(1,1)}$, the rest is analogous. Define

$$g(a,b,P_N) ::= \lambda_2(A \cap [0,a] \times [0,b]) - \frac{1}{N} m(A \cap [0,a] \times [0,b], P_N) .$$

It follows readily that

$$\begin{aligned}\left|\int_A f^{(1,1)}(x)dx - \frac{1}{N}\sum_{\substack{i=0\\ x_i\in A}}^{N-1} f^{(1,1)}(x_i)\right| &= \left|\int_0^1\int_0^1 g(a,b,P_N)\frac{\partial^2 f(a,b)}{\partial a\,\partial b}da\,db\right| \\ &\le \max_{0\le a,b\le 1}|g(a,b,P_N)|\cdot V^{(1,1)}(f) \\ &\le D(P_N,\mathcal{A}^{(1,1)})\cdot V^{(1,1)}(f)\ .\end{aligned}$$

In computer graphics the surfaces are mostly textured in order to appear realistic. In a pixel's geometry the restriction of $L(u,v)$ to a C_k is such a texture function. In most cases these are interpolation polynomials. If we consider the special situation that

$$f(u,v) = \alpha u + \beta v + \gamma \tag{5}$$

is linear, which can often be assumed in computer graphics, then we get

Corollary 4.1 *If f is linear, of the form (5), then*

$$\begin{aligned}\left|\int_A f(x)dx - \frac{1}{N}\sum_{\substack{i=0\\ x_i\in A}}^{N-1} f(x_i)\right| &\le |\alpha+\beta+\gamma|\cdot D(P_N,\mathcal{A}) \\ &\quad +|\alpha|\cdot D(P_N,\mathcal{A}^{(1,0)}) + |\beta|\cdot D(P_N,\mathcal{A}^{(0,1)})\ .\end{aligned}$$

Since we scale the pixel to the unit square, we can assume that the function $L(u,v)$ is (aside from discontinuity jumps) slowly varying. Thus, we can consider $|\alpha|$ and $|\beta|$ to be small, as compared to $|\gamma|$. Hence we are left with the discrepancy as a measure of the quality of oversampling patterns for concrete geometries represented by $\mathcal{A}$. So two directions of analysis arise:

1. Use sets of points with low discrepancy, in particular those obtained by the basic known constructions (see the next section) and

2. compute discrepancies of concrete oversampling patterns, which are being used (see the section 7).

Direction 2 was first suggested by Shirley [SHI91b], who computed the star and the extreme discrepancy for certain sampling strategies.

5 Low Discrepancy Sequences

The analysis of the previous section showed that we need point sequences with low discrepancy. In what follows we shortly review the basic ways of constructing such sequences.

We define Φ as the radical inverse function:

$$\Phi_b(i,\sigma) = \sum_{j=0}^{\infty} \sigma(a_j(i))\, b^{-j-1} \text{ when } i = \sum_{j=0}^{\infty} a_j(i)\, b^j$$

where the natural number $b > 1$ is the base, $i \in I\!N$, and σ is a permutation of the set $\{0, 1, \ldots, b-1\}$. The values Φ_b are always in the unit interval $[0, 1)$. To get an impression, how Φ acts on an i we assume σ to be the identity. Then we can imagine $\Phi_2(i)$ simply by rewriting the binary representation of i mirrored at the decimal point [1].

The Halton and the Hammersley sequence are s-dimensional vectors built out of radical inverse functions in relatively prime bases b_j, so as to avoid correlations between the components. For $0 \le i < N$ we have:

$$\text{Halton points:} \quad x_i = (\Phi_{b_1}(i), \ldots, \Phi_{b_s}(i))$$

$$\text{Hammersley points:} \quad x_i = (\frac{i}{N}, \Phi_{b_1}(i), \ldots, \Phi_{b_{s-1}}(i))$$

The points all lie inside the s-dimensional unit cube $B = [0, 1)^s$, because of the properties of the radical inverses. If the permutations are not identities, the sequences are called scrambled. Scrambling is necessary, since the radical inverse function has sequences of b_j equidistant values spaced by $\frac{1}{b_j}$. These sequences result in pieces of lines in the multidimensional case. These pieces are prone to aliasing and are eliminated by scrambling. The discrepancies $D^*(P_N)$ for both sets are bounded by (see [NIE92]):

$$D^*_{Halton} < \frac{s}{N} + \frac{1}{N} \prod_{j=1}^{s} \left(\frac{b_j - 1}{2 \log b_j} \log N + \frac{b_j + 1}{2} \right) \tag{6}$$

$$\Rightarrow \quad D^*_{Halton} \in O\left(\frac{\log^s N}{N} \right)$$

$$D^*_{Hammersley} < \frac{s}{N} + \frac{1}{N} \prod_{j=1}^{s-1} \left(\frac{b_j - 1}{2 \log b_j} \log N + \frac{b_j + 1}{2} \right) \tag{7}$$

$$\Rightarrow \quad D^*_{Hammersley} \in O\left(\frac{\log^{s-1} N}{N} \right)$$

[1] For fast algorithms for computing Φ we refer to [STR93] or [HAL64]

- *The 2-dimensional Hammersley Point Set*: We now consider the Hammersley point set in two dimensions ($s = 2$):

$$\begin{aligned} x_i &= \frac{i}{N} \\ y_i &= \Phi_b(i) \end{aligned}$$

The first component simply scans the unit square in x-direction, whereas the second component behaves randomly at first sight. The fastest generation of this sequence can be performed for base $b = 2$ without scrambling. Another reason for choosing this base is the low discrepancy of the sequence for this case. No values of the set need to be precomputed, since all values are directly computable. Increasing the sampling rate N requires to discard all previous sampling positions, therefore this pattern is not incremental and hence not well suited for adaptive oversampling. For base $b = 2$ the above bound results in

$$\begin{aligned} D^*(P_N) &< \frac{1}{2N}(\log_2 N + 7) \\ &\Rightarrow D^* \in O\left(\frac{\log N}{N}\right) \end{aligned}$$

For arbitrary bases and dimensions, stronger bounds for D^* can be found in [FAU86]. Exact values for dimension two are contained in [DCL86].

- *The 2-dimensional Halton Point Set*: For the Halton set the bases $b_1 = 2$ and $b_2 = 3$ are chosen. The sampling pattern is

$$\begin{aligned} x_i &= \Phi_2(i) \\ y_i &= \Phi_3(i) \end{aligned}$$

This pattern is incremental, meaning that increasing the number of samples is possible without discarding the samples already drawn.

In contrast to the extreme and star-discrepancy, little is known about other types of discrepancy. We have the following inequalities between the different types of discrepancies (see [NIE92]):

$$\begin{aligned} D^*(P_N) &\leq D(P_N) \leq 2^s D^*(P_N) \\ D(P_N) &\leq J(P_N) \leq 4sD(P_N)^{1/s} \end{aligned}$$

and consequently for $s = 2$

$$D(P_N, \mathcal{T}) \leq J(P_N) \leq 16\sqrt{D^*(P_N)}\,. \tag{8}$$

Since any quadrangle splits into two triangles, we also have

$$\begin{aligned} D(P_N, \mathcal{E}) &\leq 2D(P_N, \mathcal{T}) \\ D(P_N) &\leq 2D(P_N, \mathcal{T}) \ . \end{aligned}$$

From (9) we get the following asymptotic estimate: There exist constants $c_1, c_2 > 0$ such that for all N

$$c_1 \frac{\log N}{N} \leq D(P_N, \mathcal{T}) \leq c_2 \sqrt{\frac{\log^x N}{N}}$$

where P_N stands for the Halton ($x = 2$) or Hammersley ($x = 1$) point set. The lower bound holds for any N-point set (because it holds for $D(P_N)$, see [SCH72]). The precise rate of $D(P_N, \mathcal{T})$ and $D(P_N, \mathcal{E})$ for the Hammersley or Halton sequences are not known. Neither is the optimal rate, i.e.

$$\inf_{P_N \subseteq [0,1]^2} D(P_N, \mathcal{A})$$

for $\mathcal{A} = \mathcal{T}$ or $\mathcal{E}$ (see [SCH69] and [BEC87] for results in this direction). From the numerical point of view the estimates above give very little. There is no way known of computing the isotropic discrepancy of a given point set. Estimating the triangular discrepancy by (8) through the star-discrepancy yield much too pessimistic results. Some numerical evidence is collected in section 7.

6 Comparison to Usual Sampling Techniques

In order to compare the Halton and Hammersley point sets to other sampling patterns, some of the commonly used patterns on the unit square are listed ($0 \leq i < N$):

- In *regular oversampling* the square is divided into $N = n \cdot m$ rectangles, whose midpoints constitute the sampling set:

$$\begin{aligned} x_i &= \frac{i \bmod n + 0.5}{n} \\ y_i &= \frac{i \operatorname{div} n + 0.5}{m} \end{aligned}$$

- The *Poisson* sampling pattern simply consists of N random positions chosen from realizations of two independent uniformly distributed on $[0, 1]$ random variables ξ and ν. This is the pure Monte Carlo sampling.

$$\begin{aligned} x_i &= \xi_i \\ y_i &= \nu_i \end{aligned}$$

- *Jittered sampling* is stratified Monte Carlo sampling and as such a combination of the first two sampling techniques. Stratification is a means to reduce the variance of the integrand. For $N = n \cdot m$ samples and ξ and ν independent random variables we have

$$\begin{aligned} x_i &= \frac{i \bmod n + \xi_i}{n} \\ y_i &= \frac{i \operatorname{div} n + \nu_i}{m} \end{aligned}$$

- *Poisson-Disc*: As investigated in [YEL83] (see also [COO84]) the natural human eye's receptor distribution found in the border area of the retina usually does not alias despite the low density of receptors there. There the receptors are distributed randomly with a minimum distance $d > 0$. This pattern is generated by fixing a distance $d > 0$. Then we generate a random sequence of uniformly distributed points on $[0,1]^2$. A point is accepted, if its distance to the already chosen points is not smaller than d, otherwise it is rejected. The process stops, if N such samples are found.

- *N-Rook*: The N-rook pattern is generated by solving the problem in chess, that N rooks must be placed on an $N \times N$ chessboard in such a way, that they cannot defeat each other in one move. Then in each occupied square a random point is chosen. So we proceed as follows: We generate a random permutation σ of size N and obtain

$$\begin{aligned} x_i &= \frac{i + \xi}{N} \\ y_i &= \frac{\sigma(i) + \nu}{N} \end{aligned}$$

  The pattern is then fully stratified in each dimension [SHI91a]. Observe that the Hammersley point set is a special case of the permutation technique for $N = 2^m$. Taking the N-rook pattern with the Hammersley permutation results in a kind of "jittered Hammersley" with similar performance.

  Our experiments show, that randomizing the Hammersley point set only in the first component results in a sampling pattern which is superior to all examined patterns. This "randomized Hammersley" point set has to be investigated further.

Obviously the Halton and Hammersley point sets have much in common with the above sampling patterns. In addition the number N of points can be chosen

Pattern	Random	Stratified	Min-Distance	N free	Storage	regular	Incremental
Poisson	•			•			•
Poisson Disk	•	•	•	•	•		(•)
Jittered	•	•					
n-Rook	•	•		•	•		
Regular		•	•			•	
Hammersley		•	•	•		(•)	
Halton		•	•	•		•	•
Scrambled Halton		•	•	•			•

Table 1: The different sampling pattern and their properties

freely and is not limited to a decomposition of N into factors. Both sets satisfy a minimum distance property.

In table 1 the properties are listed. The patterns are classified by type (random or deterministic), stratification (implied by minimum distance or partition of the integration domain), free choice of the sample number N or restriction to a decomposition of N into factors, the need of precalculation and storage (for permutations for example), regularly spaced components, which are prone to aliasing and the property of being incremental. Table 3 shows some of the sampling patterns for $N = 64$.

7 Numerical Evidence About Discrepancies

Here we present the results of some computational experiments carried out in connection with the analysis of the previous sections. First we consider the triangular discrepancy. In table 2 we computed the theoretical bound based on (7) and (8), and compared it with numerical estimates. The latter were carried out by computing the maximal integration error over a large number of randomly chosen triangles. This should give at least an idea about the behaviour of $D(P_N, \mathcal{T})$. To get an impression of the approximation of the true maximum by the random procedure, we show results corresponding to two different numbers of triangles.

For all patterns mentioned in section 5 and 6, we compare the triangular discrepancy, the random edge discrepancy and the star discrepancy in table 4. For all patterns we used the same 100.000 random triangles (edges or domains of the form $[0, \xi] \times [0, \nu]$, respectively). For the patterns, which involve randomness themselves, a range over 10 experiments is given so that the possible variance is shown. The discrepancies are marked by a tilde to emphasize that the results are only approximations.

N	10000 random triangles	100000 random triangles	theoretical estimate
4	0.539712	0.591708	16.971
16	0.18326	0.230355	9.381
64	0.0660696	0.0777368	5.099
256	0.032454	0.0364673	2.739
1024	0.0118695	0.0178952	1.458
4096	0.00521621	0.00715305	0.771

Table 2: Triangular discrepancy of the Hammersley point set

8 The QMC-Buffer and Further Numerical Results

The quasi-Monte Carlo buffer is a simple extension of the accumulation buffer [HAE90]. This buffer first was thought for Monte Carlo integration. For every pixel of the image there exists a color variable, in which the samples are summed up and averaged afterwards. The use of the same sampling pattern over all pixels allows to apply fast buffer techniques such as z-buffer or item-buffer, which are often even available from the display's hardware at real time speed.

We replace the patterns used in [HAE90] by low discrepancy point sets. Now we use the Halton set. The QMC-buffer algorithm then works as follows:

1. For a base oversampling rate N_1 use the Halton point set for generating an image using the accumulation-buffer.

2. For single pixels, whose variance is too high, perform a further oversampling using x_i for $i > N_1$.

Since the Halton sequence is incremental, no samples have to be discarded if adaptive oversampling (see e.g. [LEE85]) is required in the second step. The first step of this algorithm can also be made adaptive: The variance over the whole image is calculated and tested by a χ^2-test. If the result is not satisfactory, we increase N_1 and go on with step 1. If the global variance is small enough, we proceed with step 2 of the algorithm.

So the QMC-buffer is an incremental algorithm which can be carried out with adaptive oversampling. This algorithm performs with a deterministic error and a rate, which is, up to the logarithm, at least as good as pure Monte Carlo integration.

Table 5 now shows a textured version of the scene and the master image I^M. In our experiments we used only simple scenes without textures. This restriction

makes sense, since through precalculation techniques like MipMapping or summed-area-tables textures can be sampled with a low variance, leaving only the variance of the geometry of a scene in the pixel. We applied the different sampling patterns to the algorithm and calculated the L_2-norm of the difference of an image I to a master image I^M(an image, which was oversampled 2500-times), since an analytical solution was not accessible. The distance between two colors C_1 and C_2 is defined as

$$d(C_1, C_2) = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}$$

where r, g, and b are in $[0, 1]$ and are the intensities of the color components red, green and blue of a pixel value. The L_2-norm of the distance of two images is now

$$\|I - I^M\|_2 = \sqrt{\frac{\sum_{\text{all pixels } i} d^2(C_i, C_i^M)}{\text{number of pixels}}} .$$

The maximum norm of the scene then is given by

$$\|I - I^M\|_\infty = \sup_{\text{all pixels } i} d(C_i, C_i^M) .$$

The results are collected in tables 6, 7 and 8[2]. Our images have a size of $512^2 = 262144$ pixels, so the maximum norm above involves the supremum over many elementary parts of the scene. Considered from this point of view, it is of interest to compare these results with the discrepancy calculations in table 4. In order to compare the L_2- and L_∞-norm with the discrepancies computed before, all values in tables 6, 7 and 8 were divided by the maximal possible $d = \sqrt{3}$. The L_2 norm gives a quality measure of the approximation of the master image by a lower sampling density.

For the randomized algorithms, we calculated 20 experiments to give an impression of the variance of these methods. For the low sampling densities (1..9) the low discrepancy sequences seem to perform bad. This seems to be caused by the origin $x_0 = (0, 0)$, which is included in the low discrepancy sets, while the other methods start with points inside the pixel. For the Halton point set we fixed this by omitting the first N' elements, resulting in the range given for that pattern. Simply omitting x_0 then performs as good as the random patterns. For the higher sampling rates a clear superiority is observed for the deterministic patterns, so x_0 can be used without disadvantages.

[2]The implementation was done on a HP9000/735. We used `drand48()` as random generator.

9 Conclusion and Further Work

We introduced the QMC-buffer for anti-aliased rendering. By applying the low discrepancy point sets, we obtain a convergence rate of about the same order as pure Monte Carlo methods. Especially applying the Halton sequence results in a very efficient incremental algorithm, which can easily be extended to adaptive pixel oversampling. By replacing randomness we obtain deterministic algorithms with a guaranteed upper error bound.

The phenomenon, that a randomized Hammersley point set always performs better than all other patterns, has to be investigated further. Moreover the optical qualities of the low discrepancy sequences have to be compared to the usual patterns by means of Fourier analysis.

Our main interest lies in solving high dimensional integral equations like the radiance equation or the complete five-dimensional description of a lens (time of aperture, size of lens and size of pixel) by the application of quasi random numbers. This is a promising effort, since the low discrepancy sequences perform well for high dimensional integrals. First results can be found in [HEI94].

References

[BEC87] J.Beck, W.W.L.Chen: Irregularities of Distribution, Cambridge University Press, 1987.

[DCL81] L.De Clerck: A proof of Niederreiter's conjecture concerning error bounds for quasi Monte Carlo integration, Adv. in Appl. Math., 2(1981), pp. 1-6.

[DCL86] L.De Clerck: A method for exact calculation of the stardiscrepancy of plane sets applied to the sequences of Hammersley, Monatsh. Math., 101(1986), pp. 261-278.

[COO84] R.L.Cook, T.Porter, L.Carpenter: Distributed ray tracing, Computer Graphics, No.3, 18(1984), pp. 137-145.

[DIP85] M.A.Z.Dippé, E.H.Wold: Antialiasing through stochastic sampling, Computer Graphics, No.3, 19(1985), pp. 69-78.

[FAU86] H.Faure: On the star-discrepancy of generalized Hammersley sequences in two dimensions, Monatsh. Math., 101(1986), pp. 291-300.

[FOL90] J.Foley, A. van Dam, S.Feiner, J.Hughes: Computer Graphics, Principles and Practice, 2nd Edition, Addison Wesley, 1990.

[HAE90] P. Haeberli, K. Akeley: The accumulation buffer: Hardware support for high-quality rendering, Computer Graphics, No.4, 24(1990), pp. 309-318.

[HAL64] J.H.Halton, G.Weller: Algorithm 247: Radical-inverse quasi-random point sequence [G5], Comm. ACM, No.12, 7(1964), pp. 701-702.

[HEI94] S.Heinrich, A.Keller: Quasi-Monte Carlo methods in computer graphics, Part II: The radiance equation, Technical Report, University of Kaiserslautern, to appear.

[LEE85] M.E.Lee, R.A.Redner, S.P.Uselton: Statistically optimized sampling for distributed ray tracing, Computer Graphics, No.3, 19(1985), pp. 61-67.

[MIT87] D.P.Mitchell: Generating antialiased images at low sampling densities, Computer Graphics, No.4, 21(1987), pp. 65-72.

[MIT91] D.P.Mitchell: Spectrally optimal sampling for distribution ray tracing, Computer Graphics, No.4, 25(1991), pp. 157-162.

[NIE72] H.Niederreiter: Application of diophantine approximations to numerical integration, in: Diophantine Approximation and its Applications, C.F.Osgood, ed., Academic Press, 1973, pp. 120-199.

[NIE78] H.Niederreiter: Quasi-Monte Carlo methods and pseudo-random numbers, Bull. Amer. Math. Soc., 84(1978), pp. 957-1041.

[NIE92] H.Niederreiter: Random Number Generation and Quasi-Monte Carlo Methods, SIAM Philadelphia, Pennsylvania 1992.

[PAI89] J.Painter, K.Sloan: Antialiased ray tracing by adaptive progressive refinement, Computer Graphics, No.3, 23(1989), pp. 281-288.

[SCH69] W.M.Schmidt: Irregularities of distribution IV, Inventiones Math., 7(1969), pp. 55-82.

[SCH72] W.M.Schmidt: Irregularities of distribution VII, Acta Arith., 21(1972), pp. 45-50.

[SHI91a] P.Shirley: Physically Based Lighting Calculations for Computer Graphics, Ph.D.Thesis, University of Illinois, 1991.

[SHI91b] P.Shirley: Discrepancy as a quality measure for sample distributions, Eurographics '91, F.H.Post and Barth (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 183-195.

[SOB91] I.M.Sobol: Die Monte-Carlo-Methode, Deutscher Verlag der Wissenschaften, Berlin 1991.

[STR93] J.Struckmeier: Fast generation of low-discrepancy sequences, Berichte der Arbeitsgruppe Technomathematik, Nr.93, Fachbereich Mathematik, Universität Kaiserslautern.

[TRA88] J.F.Traub, G.W.Wasilkowski, H.Woźniakowski: Information-Based Complexity, Academic Press, 1988.

[WAT93] A.Watt, M.Watt: Advanced Animation and Rendering Techniques, Theory and Practice, Addison Wesley, 1993.

[WOZ91] H.Woźniakowski: Average case complexity of multivariate integration, Bull. Amer. Math. Soc., 24(1991), pp. 185-194.

[YEL83] J.I.Yellot, Jr.: Spectral consequences of photoreceptor sampling in the rhesus retina, Science, 221(1983), pp. 382-385.

[ZAR70] S.K.Zaremba: La discrépance isotrope et l'intégration numérique, Ann. Mat. Pura Appl., 87(1970), pp. 125-136.

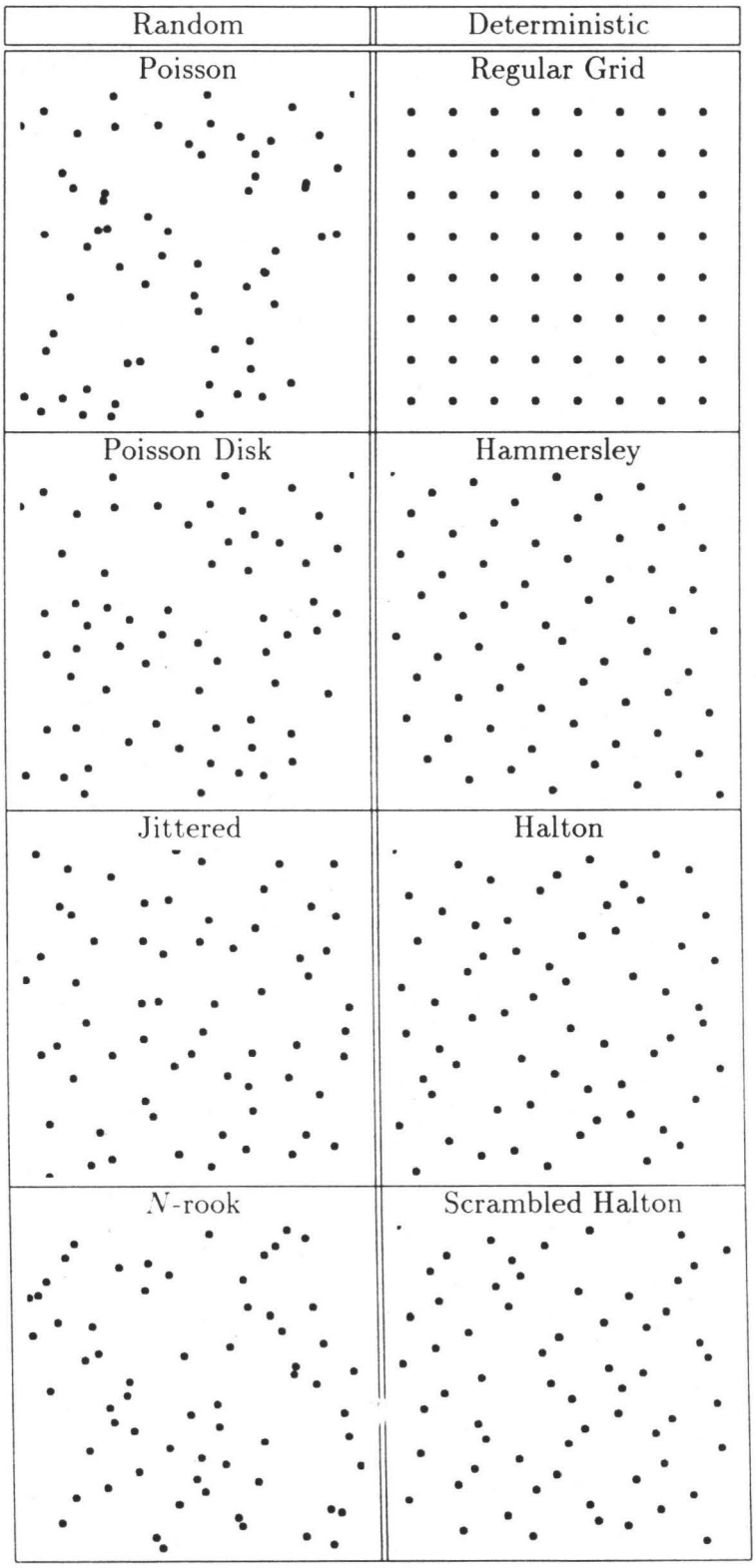

Table 3: The different sampling patterns for $N = 64$

Pattern	N	$\check{D}(P_N, \mathcal{T})$	$\check{D}(P_N, \mathcal{E})$	$\check{D}^*(P_N)$
Poisson	4	0.590735...0.909109	0.424773...0.728038	0.439906...0.703657
	16	0.269943...0.475035	0.205515...0.35258	0.19317...0.357886
	64	0.136628...0.189286	0.124042...0.162481	0.108418...0.171163
	256	0.0702834...0.104564	0.0469791...0.10384	0.0545187...0.081455
	1024	0.0307715...0.0573265	0.0310258...0.0594817	0.0243101...0.0567063
Poisson Disk	4	0.522657...0.765606	0.380949...0.618871	0.351455...0.622014
	16	0.212796...0.325906	0.186497...0.319828	0.188717...0.250398
	64	0.0903901...0.115629	0.0749705...0.135366	0.0692996...0.100529
	256	0.0389122...0.0639217	0.0373723...0.0766956	0.0341753...0.0503391
	1024	0.0191191...0.0328588	0.0157817...0.0308078	0.0154991...0.0304187
Jittered	4	0.49858...0.710208	0.331581...0.498499	0.332042...0.588175
	16	0.215582...0.277194	0.152869...0.221958	0.182205...0.239541
	64	0.0924101...0.123592	0.0628123...0.0840382	0.0702102...0.0972717
	256	0.0330829...0.0420403	0.0240844...0.0315302	0.0247497...0.0363527
	1024	0.0135303...0.0159187	0.0091878...0.0105786	0.0105643...0.013188
N-rook	4	0.450272...0.849446	0.354266...0.461543	0.321364...0.468145
	16	0.227336...0.324756	0.155742...0.280876	0.119916...0.214904
	64	0.109728...0.145775	0.0807091...0.139889	0.0605287...0.107181
	256	0.0507363...0.0796776	0.0447672...0.0609385	0.0254601...0.0479083
	1024	0.0276874...0.0474817	0.0234402...0.0330985	0.0141554...0.0245528
Regular Grid	4	0.559198	0.248758	0.434216
	16	0.231485	0.124932	0.226729
	64	0.102398	0.060865	0.112807
	256	0.0432194	0.0310497	0.0607353
	1024	0.0171027	0.0101752	0.0304701
Hammersley	4	0.591708	0.466508	0.498039
	16	0.230355	0.181719	0.170741
	64	0.0777368	0.0742586	0.0520124
	256	0.0364673	0.0349363	0.0146222
	1024	0.0178952	0.0149017	0.0040708
Halton	4	0.695907	0.594563	0.497245
	16	0.235361	0.21432	0.199702
	64	0.116366	0.106892	0.0481554
	256	0.0371292	0.0311993	0.0168317
	1024	0.0139491	0.012062	0.0056194
Scrambled Halton	4	0.613899	0.471222	0.497245
	16	0.283306	0.211999	0.159189
	64	0.10833	0.0755593	0.0563067
	256	0.0352614	0.033048	0.018281
	1024	0.0142124	0.0112995	0.00543235

Table 4: Discrepancies of patterns

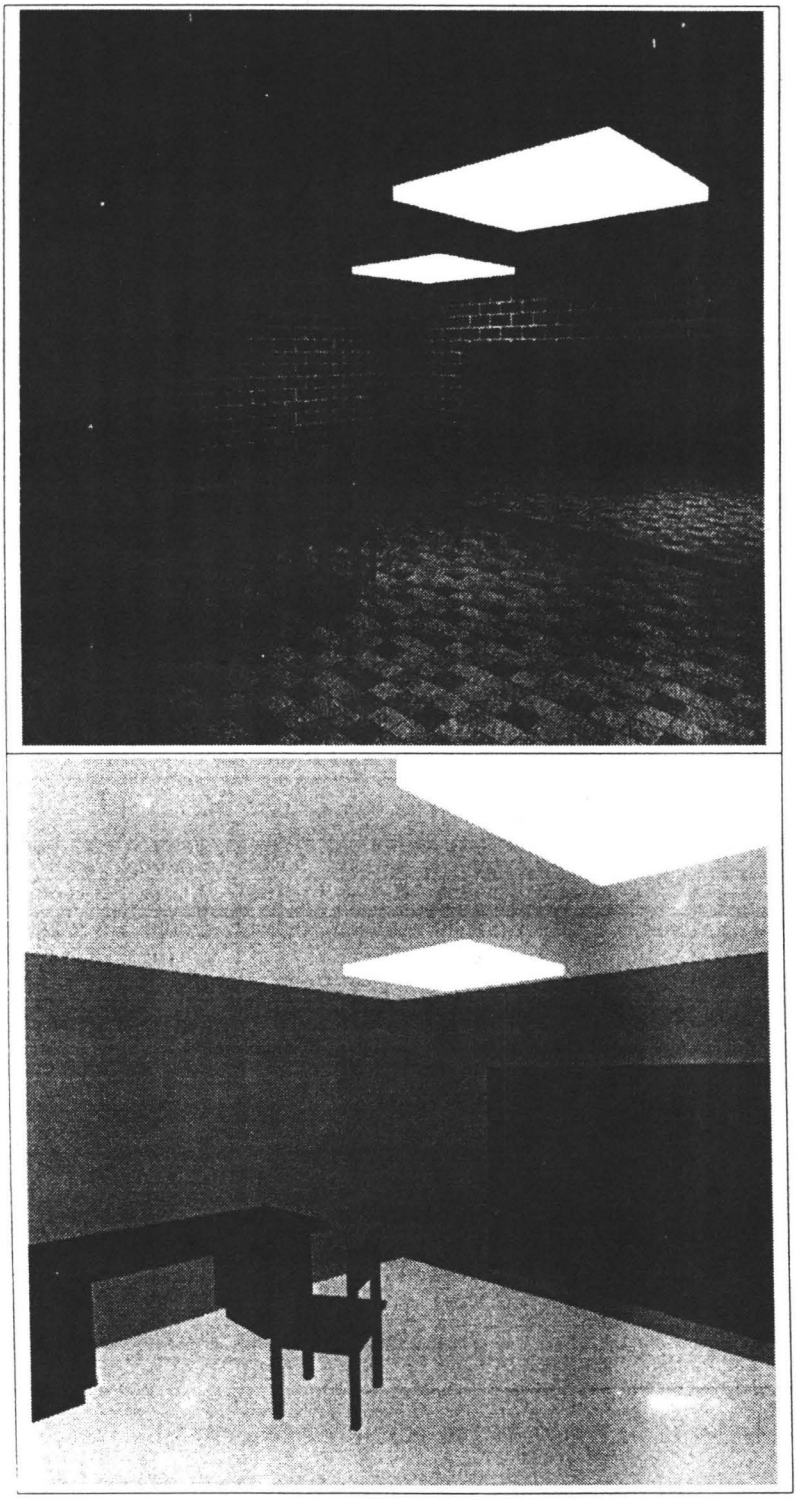

Table 5: Textured image and master image

Pattern	N	1	2	3	4	5
Poisson	L_2	0.0170...0.0286	0.0118...0.0218	0.0093...0.0226	0.0078...0.0194	0.0077...0.0187
	L_∞	0.3917...0.7041	0.3419...0.5547	0.2875...0.5140	0.2853...0.4913	0.2694...0.4913
Poisson Disk	L_2	0.0170...0.0286	0.0099...0.0154	0.0078...0.0149	0.0062...0.0119	0.0051...0.0104
	L_∞	0.3917...0.7041	0.2672...0.4030	0.2468...0.3781	0.1947...0.3532	0.1811...0.3215
Jittered	L_2	0.0170...0.0286	0.0113...0.0217		0.0072...0.0107	
	L_∞	0.3917...0.7041	0.3464...0.5547		0.2128...0.3464	
N-rook	L_2	0.0170...0.0268	0.0105...0.0150	0.0077...0.0132	0.0062...0.0074	0.0052...0.0066
	L_∞	0.3917...0.6951	0.3147...0.3917	0.2604...0.3645	0.2060...0.2785	0.1789...0.3215
Regular Grid	L_2	0.0169	0.0132		0.0072	
	L_∞	0.3917	0.3645		0.2332	
Hammersley	L_2	0.0321	0.0169	0.0154	0.0091	0.0107
	L_∞	0.7064	0.3917	0.3645	0.2445	0.2740
Halton	L_2	0.0177...0.0321	0.0100...0.0185	0.0080...0.0154	0.0058...0.0113	0.0055...0.0122
	L_∞	0.3985...0.7064	0.2853...0.3985	0.2491...0.3645	0.2060...0.3215	0.1947...0.3192
Randomized Hammersley	L_2	0.0170...0.0286	0.0101...0.0163	0.0081...0.0129	0.0057...0.0085	0.0052...0.0104
	L_∞	0.3917...0.6973	0.2491...0.3917	0.2604...0.3668	0.1857...0.2445	0.1857...0.2287

Table 6: L_2-norm and maximum error, Part I

Pattern	N	6	7	8	9
Poisson	L_2	0.0070...0.0150	0.0055...0.0144	0.0056...0.0159	0.0047...0.0154
	L_∞	0.2377...0.4823	0.1789...0.4030	0.1653...0.4189	0.1630...0.3532
Poisson Disk	L_2	0.0047...0.0096	0.0044...0.0081	0.0043...0.0069	0.0038...0.0069
	L_∞	0.1789...0.3124	0.1540...0.2830	0.1426...0.2196	0.1517...0.2377
Jittered	L_2	0.0048...0.0092			0.0042...0.0061
	L_∞	0.1653...0.3306			0.1404...0.2762
N-rook	L_2	0.0048...0.0061	0.0041...0.0051	0.0037...0.0048	0.0033...0.0045
	L_∞	0.1766...0.2491	0.1426...0.2377	0.1313...0.2128	0.1155...0.1970
Regular Grid	L_2	0.0053			0.0046
	L_∞	0.2287			0.1992
Hammersley	L_2	0.0082	0.0079	0.0053	0.0066
	L_∞	0.2287	0.2287	0.2083	0.2174
Halton	L_2	0.0047...0.0087	0.0046...0.0090	0.0038...0.0059	0.0035...0.0066
	L_∞	0.1585...0.2423	0.1585...0.2423	0.1177...0.2287	0.1291...0.2332
Randomized Hammersley	L_2	0.0042...0.0078	0.0041...0.0075	0.0037...0.0050	0.0036...0.0063
	L_∞	0.1743...0.2445	0.1426...0.2287	0.1155...0.2128	0.1381...0.2332

Table 7: L_2-norm and maximum error, Part II

Pattern	N	15	16	25	36	64
Poisson	L_2	0.0037...0.0102	0.0036...0.0086	0.0024...0.0071	0.0029...0.0065	0.0016...0.0050
	L_∞	0.1404...0.2808	0.1381...0.2694	0.1223...0.2196	0.1109...0.2083	0.0725...0.1834
Poisson Disk	L_2	0.0029...0.0056	0.0028...0.0041	0.0022...0.0040	0.0019...0.0034	0.0010...0.0022
	L_∞	0.1064...0.2038	0.1041...0.1811	0.0883...0.1540	0.0657...0.1291	0.0408...0.1064
Jittered	L_2		0.0029...0.0040	0.0024...0.0030	0.0020...0.0024	0.0011...0.0014
	L_∞		0.1132...0.1925	0.0815...0.1336	0.0611...0.1313	0.0430...0.0792
N-rook	L_2	0.0025...0.0037	0.0025...0.0036	0.0020...0.0029	0.0020...0.0025	0.0010...0.0017
	L_∞	0.0996...0.2060	0.1064...0.2106	0.0725...0.1721	0.679...0.1743	0.0475...0.0974
Regular Grid	L_2		0.0033	0.0026	0.0022	0.0014
	L_∞		0.1313	0.1404	0.1041	0.0770
Hammersley	L_2	0.0043	0.0032	0.0031	0.0022	0.0010
	L_∞	0.1494	0.1087	0.1132	0.0860	0.0521
Halton	L_2	0.0029...0.0047	0.0025...0.0038	0.0021...0.0036	0.0018...0.0024	0.0010...0.0012
	L_∞	0.1041...0.2015	0.0951..0.1766	0.0725...0.1336	0.0589...0.0815	0.0430...0.0747
Randomized Hammersley	L_2	0.0024...0.0041	0.0026...0.0030	0.0019...0.0030	0.0017...0.0022	0.0009...0.0010
	L_∞	0.0906...0.1811	0.0770...0.1313	0.0657...0.1223	0.0498..0.0928	0.0385...0.0566

Table 8: L_2-norm and maximum error, Part III