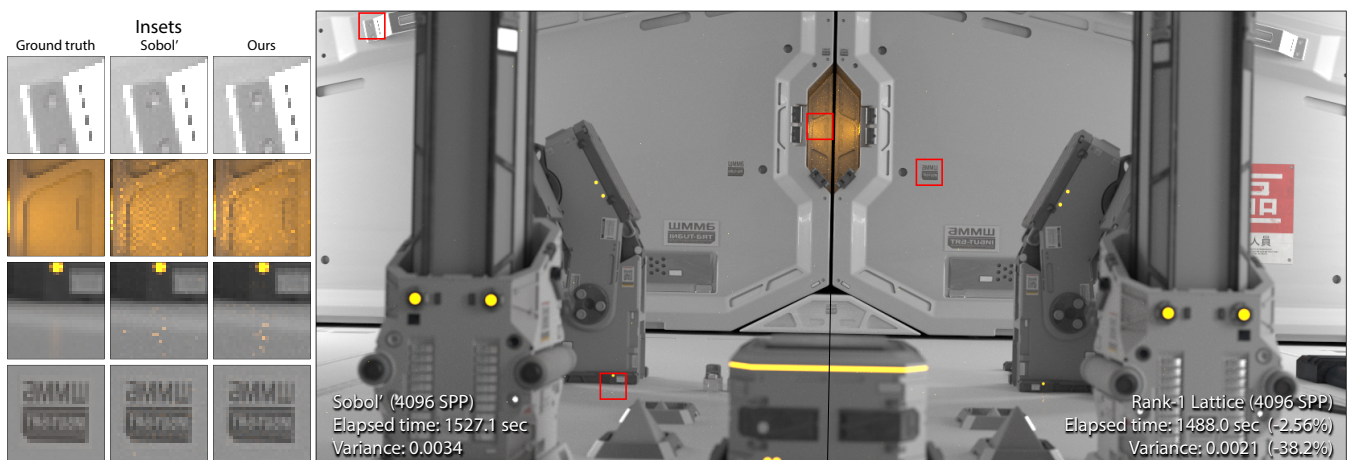# Rank-1 Lattices for Efficient Path Integral Estimation

Hongli Liu[1] and Honglei Han[1] [†] and Min Jiang[2]

[1]State Key Laboratory of Media Convergence and Communication,
and School of Animation and Digital Arts, Communication University of China, China
[2]Framestore, United Kingdom

**Figure 1:** *Quasi-Monte Carlo in ray-traced renderings requests high-dimensional robust quasi-random sequences, which are normally impossible to achieve by general rank-1 lattices. Our framework produces rank-1 lattices particularly optimized for estimating the path integral. It successfully achieves promising rendered results compared to the Sobol′ sequence.*

**Abstract**

*We introduce rank-1 lattices as a quasi-random sequence to the numerical estimation of the high-dimensional path integral. Previous attempts at utilizing rank-1 lattices in computer graphics were very limited to low-dimensional applications, intentionally avoiding high dimensionality due to that the lattice search is NP-hard. We propose a novel framework that tackles this challenge, which was inspired by the rippling effect of the sample paths. Contrary to the conventional search approaches, our framework is based on recursively permuting the preliminarily selected components of the generator vector to achieve better pairwise projections and minimize the discrepancy of the path vertex coordinates in scene manifold spaces, resulting in improved rendering quality. It allows for the offline search of arbitrarily high-dimensional lattices to finish in a reasonable amount of time while removing the need to use all lattice points in the traditional definition, which opens the gate for their use in progressive rendering. Our rank-1 lattices successfully maintain the pixel variance at a comparable or even lower level compared to Sobol′ sampler, which offers a brand new solution to design efficient samplers for path tracing.*

**CCS Concepts**

• *Computing methodologies* → *Rendering;*

## 1. Introduction

In computer graphics, *quasi-Monte Carlo* (QMC) methods play a vital role in the numerical estimation of complicated integrals,

where in particular, the simulation of photorealistic light transport in virtual scenes [Vea98] profits from QMC point sampling using low-discrepancy sequences (LDS). The problem of path integral estimation lies in finding paths connecting the pixels and the light

[†] Corresponding author: hanhonglei@cuc.edu.cn

sources. Among the sampling strategies, path tracing and bidirectional path tracing are the most popular ones, where the paths are produced randomly using random sampling or LDS. LDS generate sets of high-dimensional points in predictable manners, and the points fill the sampling hypercube more evenly than the random sampled ones. The integration usually converges quadratically faster [KK02], and less noisy images could be rendered in shorter time. Digital nets and lattice rules are two LDS that were developed in parallel and widely used [OG18]. Rank-1 lattices are ones of the most popular lattice rules. They are sequences of vectors distributed in unit hypercubes, with each vector corresponding to a unique sample index. Since Dammertz introduced rank-1 lattices to computer graphics [Dam09], the attention of scholars has focused on their applications in 2-D sampling [MBR*13], grid-based simulation [DKD08], and other low-dimensional cases because of its two distinct advantages. First, large mutual minimum distances can be easily achieved in lower dimensions (especially 2-D and 3-D). Second, due to their simple congruential forms, their runtime computational cost is low, which results in a reduced computational time and engineering complexity.

However, the rank-1 lattices are not as robust as digital nets [LM16], especially in high dimensionality. This to some extent makes them less popular in high-dimensional numerical estimation. To the best of our knowledge, rank-1 lattices have never been adopted in the field of path integral estimation. First, by definition, rank-1 lattices postulate using all sample points, which reduces their practical application, especially in progressive rendering that claims a dynamic sample count. Second, finding a good rank-1 lattice of high dimensionality is a known NP-hard problem with exponentially growing complexity [Car02, Hel85] (more details will be followed in Sec. 3.1). In some cases like rendering a volumetric scattering material concerning thousands of internal bounces of the light path, the existing methods are unable to find such a high-dimensional lattice. Furthermore, high-dimensional rank-1 lattices inevitably suffer from correlation issues with arbitrary subsets of lower-dimensional projections.

To adopt rank-1 lattices in path tracing, we investigated the integration process. We noticed one thing that differed the path integrals from general multi-dimensional integrals was that the integration variables, i.e., the path vertices at each path length, were correlated. For example, consider a path of length 2. Modifying the first vertex position results in the change of the second vertex position, even if the (quasi-) random vector to generate the second vertex remains unchanged. This is known as the rippling effect: the continuing and spreading results of a change. In this perspective, the (quasi-) random vector is actually decorrelated to the path vertices. Therefore, even when the distributed sample vectors is not well designed, they might still generate decorrelated vertices.

With the above observation, we present a novel solution to adopt rank-1 lattices in estimating path integrals. In Sec. 3, we analyzed the computational challenge of the lattice search algorithms in high dimensionality. To seek another solution, we examined the correlation between the quasi-random vectors and the discrepancy of intersection positions in the scene manifold space with an example. By simply permuting an ill-formed lattice into improved 2-D projections, more randomness (decorrelation) of the manifold posi-

tions was achieved, thanks to the rippling effect. The rippling effect guarantees an enhanced decorrelation between the vertices and the input random vectors. We provide evidence by differentiating the vertex positions to support our theory. In Sec. 4, we formulate a framework making use of the rippling effect. It involves two steps, first finding the best preliminary components, then performing pairwise permutation. Finally, in Sec. 5, we verified our framework by a series of experiments. In addition to a lattice search time comparison against the widely used *component-by-component* (CBC) algorithm [KJ02], we compare the sampling efficiency mainly with the Sobol′ sequence [Sob67], which is widely accepted to be dimentsionally robust. The experimental results prove the effectiveness of the lattice-based samplers. Our framework creates the possibility of constructing arbitrarily high-dimensional rank-1 lattices for rendering.

Our contributions are:

- Proof-of-concept of rank-1 lattices being worthy candidates in (bidirectional) path tracing as quasi-random sequences, inspired by the rippling effect of the sample path;
- Novel criterion that efficiently measures the quality of incomplete lattice sequences;
- Solid lattice-search framework optimized for arbitrarily high dimensionality;
- Lattice solution for designing high-efficiency samplers for rendering applications.

## 2. Preliminaries

In this section, we describe the most relevant theories and methods to draw the blueprint of our approach.

**Rank-1 lattices.** The research of rank-1 lattices was pioneered by Korobov and examined by many others [Dam09]. The rank-1 lattice rule is a quadrature rule for estimating the integral over the unit $s$-dimensional hypercube for a function $f$, in the form

$$E(f) \approx \frac{1}{n} \sum_{i=0}^{n-1} f(\mathbf{x}_i), \qquad (1)$$

where $\mathcal{L}_n = (\mathbf{x}_0, \dots, \mathbf{x}_{n-1})$ is the rank-1 lattice with points

$$\mathbf{x}_i = \left\{ \frac{i}{n} \mathbf{g} \right\}_1 \in [0,1)^s, \; i = 0, \dots, n-1, \qquad (2)$$

$n$ is the total number of points, $\mathbf{g} = (g_1, \dots, g_s)$ is the generator vector of integer components, and $\{\mathbf{x}\}_1$ denotes the fractional part of vector $\mathbf{x}$. A valid generator vector to fulfill the condition is:

$$\gcd(g_1, \dots, g_s, n) = 1. \qquad (3)$$

**Lattice search algorithms.** As indicated in Eq. 2, the distribution quality of a rank-1 lattice is governed by the generator vector $\mathbf{g}$. Given a quality criterion, the lattice search is equivalent to selecting the components of $\mathbf{g}$ from all possible candidates to maximize (or minimize) the criterion.

While the quality of a digital net is measured by their discrepancy [Nie87, OG18], a rank-1 lattice is usually measured by the mutual minimum distance

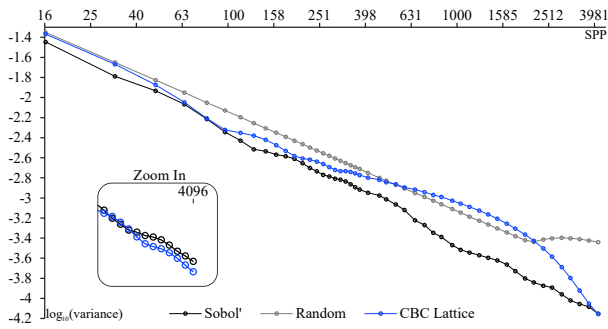$$d_{min}(\mathcal{L}_n) = \min_{0 \le i < j < n} \|\mathbf{x}_i - \mathbf{x}_j\| \qquad (4)$$

**Figure 2:** *Variance in rendering of the Cornell Box scene.*



**Figure 3:** *The best and worst projections of a 12-D lattice ($n = 1024$) search by the CBC algorithm.*



**Figure 4:** *A sample path generated by importance sampling.*

using the Euclidean distance [Dam09]. Several methods have been developed to efficiently find the minimum distance, including the *spectral test* [Llo82], *Lenstra-Lenstra-Lovász (LLL) Basis Reduction* [Bre11], and exhaustively searching the distance to the origin [Dam09]. The search algorithm is then adopted to find the components of the generator vector, by maximizing the overall minimum distance or the ones of lower-dimensional projections. The mathematical software *Lattice Builder* was introduced by L'Ecuyer and Munger [LM16] which is a comprehensive tool for building lattices, containing the implementation of exhaustive and random searches, as well as CBC and random CBC constructions.

**Limitation.** The brevity of rank-1 lattices leads to the lack of robustness of the rank-1 lattices, which are mainly because of two constraints.

**Constraint 1: using all sample points.** Using only part of the sample points potentially omits large areas of the sampling hypercube [OG18], which results in a considerable bias. This is because traditional approaches to search for a rank-1 lattice never consider an incomplete sequence of lattice points, as it adds a large expense to the existing computational cost. To illustrate this, we measured the average variances of the renderings using the Sobol′ sequence, random sampling, and a rank-1 lattice constructed using the CBC algorithm in *Lattice Builder*, with dimension $s = 16$ and interval count $n = 4096$. As plotted in Fig. 2, the lattice produced poor convergence even worse than random sampling when using incomplete sequences of sample points. Such samplers are not practical in production, especially where the artists terminate the sampling process at a random point in time.

**Constraint 2: low dimensionality.** The utilization of rank-1 lattices is prevented in two ways in high-dimensional integration. On the one hand, as the number of dimension increases, it becomes impossible to guarantee the evenness of the projection of every pair of dimensions, i.e., ill-formed projections are inevitable even when enumerating all possible generator vectors of such dimensionality (see Fig. 3). On the other hand, searching such a generator vector has an extremely undesirable exponential computational cost using the existing approaches, making them impossible to terminate in finite time.

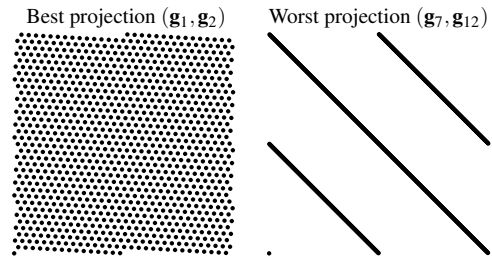Given this, our motivation is to start with a featured characteristic
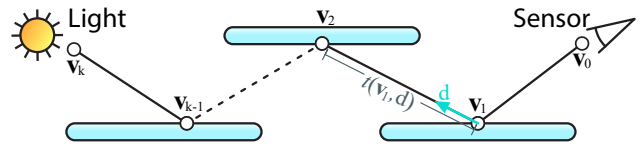
of path integrals, the rippling effect which makes them distinct from other general integrals.

**Path integral of light transport.** The path integral framework [Vea98] measures the camera response in the form of an integral:

$$I = \int_{\Omega} f(\overline{p}) \, d\mu(\overline{p}) \qquad (5)$$

where $\overline{p} = \mathbf{v}_0 \ldots \mathbf{v}_k \in \Omega$ is a complete path from a light to the camera, with all points on the scene manifold; $\Omega$ is the set of all possible light paths of arbitrary length; $d\mu(\overline{p}) = dA(\mathbf{v}_0) \ldots dA(\mathbf{v}_k)$ is the area product measure, and $f$ is the measurement contribution function.

**Importance sampling.** As one of the most useful schemes to efficiently sample the path integral, importance sampling in path tracing refers to the principle of choosing a density function $p$ proportional to the reflectance function (e.g., the *bidirectional reflectance distribution function* (BRDF)). In local sampling, the next vertex of a given incomplete path is determined by the previous vertices

$$\mathbf{v}_{i+1} = \mathbf{v}_i + t(\mathbf{v}_i, \mathbf{d}) \cdot \mathbf{d} \qquad (6)$$

where $\mathbf{d}$ is the sampling direction chosen from the inverse cumulative distribution function of $p$ at $\mathbf{v}_i$ viewing from $\mathbf{v}_{i-1}$ (for convenience, we directly use $\mathbf{v}_i$ as a notation of its position); $t(\mathbf{v}, \mathbf{d})$ measures the distance to the next closest intersection at $\mathbf{v}$ in direction $\mathbf{d}$, as illustrated in Fig. 4. It's worth mentioning that a unified directional vector has a degree of freedom of 2 (usually in the spherical coordinate), which indicates that $\mathbf{x}$ is (up to) 2-D for each scene vertex.

**The Rippling effect.** The phase of "rippling effect" was first discussed in the course of the research on *Markov Chain Monte Carlo* (MCMC) algorithms, stating that subtle modification of a path vertex results in a large change to the subsequent path. Support for this conclusion lies in the dependency of adjacent vertices in Eq. 6, but lacked in-depth assessments in previous work. In later sections, we

shall give both intuitive and stringent explanations of the substance of the rippling effect and how our framework relies on this effect to specifically resolve the dimensionality issue of rank-1 lattices for path integral estimation.

## 2.1. Related work

**Lattices in Korobov forms.** Korobov rank-1 lattices $\mathcal{L}_{n,a}$ are a special class of rank-1 lattices whose generator vector has form as $\mathbf{g} = (1, a, a^2, \ldots, a^s)$ . While naively searching a high-dimensional rank-1 lattice is hard, searching one in Korobov form is relatively easy, since it only contains a single parameter $a$. However, in our paper, we didn't consider the use of Korobov lattices due to two reasons. First, Korobov lattices are not able to solve the correlation issue of high-dimensional sample points as shown in Fig. 3. Secondly, they only use a single parameter, which makes them less controllable, especially when considering incomplete sequences.

**Concatenated 2-D Sequences.** Since well-distributed 2-D rank-1 lattices are easy to obtain, one straightforward approach to construct high-dimensional lattice-based samplers is to concatenate multiple 2-D sequences by shuffling the points [CKK18, PJH17b].The reason why we insist on optimizing high-dimensional rank-1 lattices is that the shuffling process usually costs an extra computation time and degrades the sampler's scalability in progressive sampling. For example, one designing a real-time ray-tracing application would not prefer shuffling due to its cost and inconvenience.

## 3. Our method: blueprint

Before describing the technical details in section 4, we provide some illustrative and intuitive explanations and introduce some concepts first. In Sec. 3.1, we describe how the time complexity of the conventional approach hinders the search for high-dimensional generator vectors. To distinguish the particular context of path integration from general multidimensional sampling, we use a special example to explain the substance of the rippling effect in Sec. 3.2.1. Then, in Sec. 3.2.2, we provide the theoretical deduction to support our inference and disclose our core contribution: the pairwise permutation framework. Finally, in Sec. 3.3, we discuss a lower bound of the minimum distance of a sequence of lattice points. A scheme of finding preliminary components is then presented to maximize this bound.

## 3.1. The crux of high dimensionality

Consider an $s$-dimensional rank-1 lattice $\mathcal{L}_n = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. The naive search algorithm [Dam09] walks through all $\frac{n!}{(n-s)!s!}$ possible permutations of the components and return the vector of the largest minimum distance (yielding the so-called *maximized minimum distance* (MMD) lattice), with a complexity of $O\left(K(s) \cdot \min\left(n^s, n^{n-s}\right)\right)$, where $K(s) = s^2 \log s$ is the average polynomial complexity of computing the minimum distance for an $s$-dimensional rank-1 lattice using *LLL basis reduction* [JSM08].

Therefore, even the MMD lattice in high dimensionality does exist, it is impractical to find using exhaustive methods. As a compromise, there are some faster approaches for finding local optimal rank-1 lattices. For example, the CBC algorithm [KJ02] that appends new components, one at a time, to the end of the generator vector, and early termination of lattice search proposed by Finckh et al. [FDL14]. Unfortunately, they can not meet the dimensionality demand mainly due to:

1. The complexity of $O(n^s)$ is still too costly.
2. The resulted lattices must use all sample points. If incomplete point sequences are considered, the existing cost will be greatly increased by

   a. enumeration of different sequence lengths being required;
   b. the fast methods for computing the minimum distance (like *LLL basis reduction*) being no longer valid as they must account for all lattice points.

Therefore, a novel framework of searching for the generator vector is required to fulfill our demand for path tracing.

## 3.2. The rippling effect

### 3.2.1. An example

In general, the complexity of reflection models and intricate visibility relationships among arbitrary scene geometry all lead to the impossibility of analytically investigating the path integral [PJH17b]. To start the investigation on the correlation of quasi-random vector and positions of the path vertices in the scene manifold, we demonstrate one specific uncomplicated setting.

As an example, consider the interior surface of a unit *Lambertian* sphere, where the density function for importance sampling is $p = \frac{\cos\theta}{\pi}$ with θ being the angle between the surface normal and ray direction. Due to symmetry, the outgoing radiance at arbitrary points on the surface must be the same. As a result, our investigation starts at a random point on the surface. We trace the vertices at different path lengths using importance sampling, record their positions, and transform the positions to the manifold space scaled to a unit square. The result is displayed in Fig. 5. To expound the contrast, we adopt three different quasi-random sequences for comparison: a deliberately chosen rank-1 lattice with each pair of projection exhibiting highly ill-formed patterns (titled "Original"); another rank-1 lattice whose components of the generator vector are identical to the first one except that they are in a different order (titled "Permuted"); and the Sobol′ sequence.

We use the notion of *star discrepancy* [Nie87, Thi01] to quantify how much the manifold positions resemble the random distribution

$$D^*(B, P) = \sup_{b \in B} \left| \frac{|\mathbf{x}_i \in b|}{N} - V(b) \right| \quad (7)$$

where $P = (\mathbf{x}_0, \ldots, \mathbf{x}_{n-1})$ is the manifold coordinates; $|\mathbf{x}_i \in b|$ is the number of points in $b$; $V(b)$ measures the volume of $b$; and $B$ is the family of rectangular box shapes in the subsets of $[0, 1)^2$. The measured discrepancies $D^*$ of all examined manifolds are labelled in the images. There are two observed facts to be noted:

1. Although the "Original" lattice exhibits highly linearly correlated patterns in lower path lengths, the discrepancy descends fast with the increase of path length. In other words, due to

| | path length = 1 | | | path length = 2 | | | path length = 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Quasi-random | Vert. position | Manifold | Quasi-random | Vert. position | Manifold | Quasi-random | Vert. position | Manifold |
| Original | | | $D^*=0.2055$ | | | $D^*=0.0622$ | | | $D^*=0.0504$ |
| Permuted | | | $D^*=0.0082$ | | | $D^*=0.0226$ | | | $D^*=0.0253$ |
| Sobol′ | | | $D^*=0.0149$ | | | $D^*=0.0223$ | | | $D^*=0.0351$ |

**Figure 5:** *Numerically integrating in the interior of a unit sphere. From left to right, the path length increases, exhibiting the change of correlation between the quasi-random sequences, traced vertex positions, and the discrepancies of corresponding manifold coordinates.*

the rippling effect, the vertices become increasingly random and decorrelated to the input lattice points. It can be justifiably inferred that, if the quasi-random patterns are decorrelated at each path length (especially initially), decorrelated manifold coordinates will occur.

2. The "Permuted" lattice, despite having the identical components as the "Original" lattice (and thus the identical distribution), obtains decorrelated manifold coordinates as well as a manifold discrepancy close to the Sobol′ sequence.

**Conclusion.** As presented, by simply permuting a deliberately chosen "bad" lattice, manifold discrepancy and patterns similar to the Sobol′ sequence can still be achieved. The decorrelation between the input sample vectors and the output manifold positions expands with the increase of path length, which is exactly the substance of the so-called rippling effect. It offers a possible solution to our lattice search, namely, to perform 2-D optimizations at each path length to lay the components of "bad" projections in different path lengths and decorrelate them into randomly distributed manifold positions.

However, this extremely simple example offers only intuitive illustrations, while having little help for general-purpose rendering. When arbitrarily complicated geometry and scene manifold, as well as different materials and optical effects are included, the correlation is unpredictable. We shall prove the correlation in the next subsection.

### 3.2.2. Generalization

**Theorem 1** In importance sampling, the change of quasi-random vectors at non-Dirac (i.e. non-specular) path vertices contributes to an increased variation of the subsequent vertices with respect to the delta path length.

Consider a function $f$ which, resembling Eq. 6, gives the manifold positions for the path vertices

$$\mathbf{v}_{i+1}^{\mathcal{M}} = f(\mathbf{v}_i^{\mathcal{M}}, \xi_i) \tag{8}$$

where $\mathbf{v}_i^{\mathcal{M}} \in \mathcal{M}$ is the position of the $i^{\text{th}}$ vertex in the 2-D scene manifold space $\mathcal{M}$; $\xi_i \in [0,1)^2$ is the $i^{\text{th}}$ 2-D quasi-random vector; the partial derivatives of $f$ with respect to the first argument is denoted by $D_1 f$, and the second argument by $D_2 f$, respectively. The special part lies in that, the vertices $\mathbf{v}_1^{\mathcal{M}}, \dots, \mathbf{v}_k^{\mathcal{M}}$ are determined by $\xi_0, \dots, \xi_{k-1}$ and the initial position $\mathbf{v}_0^{\mathcal{M}}$.

We unfold the total differential for the manifold positions

$$\begin{aligned}
\mathrm{d}\mathbf{v}_1^{\mathcal{M}} =& D_1 f(\mathbf{v}_0^{\mathcal{M}}, \xi_0)\mathrm{d}\mathbf{v}_0^{\mathcal{M}} + D_2 f(\mathbf{v}_0^{\mathcal{M}}, \xi_0)\mathrm{d}\xi_0 \\
\mathrm{d}\mathbf{v}_2^{\mathcal{M}} =& D_1 f(\mathbf{v}_1^{\mathcal{M}}, \xi_1)\mathrm{d}\mathbf{v}_1^{\mathcal{M}} + D_2 f(\mathbf{v}_1^{\mathcal{M}}, \xi_1)\mathrm{d}\xi_1 \\
=& D_1 f(\mathbf{v}_1^{\mathcal{M}}, \xi_1)D_1 f(\mathbf{v}_0^{\mathcal{M}}, \xi_0)\mathrm{d}\mathbf{v}_0^{\mathcal{M}} \\
& + D_1 f(\mathbf{v}_1^{\mathcal{M}}, \xi_1)D_2 f(\mathbf{v}_0^{\mathcal{M}}, \xi_0)\mathrm{d}\xi_0 + D_2 f(\mathbf{v}_1^{\mathcal{M}}, \xi_1)\mathrm{d}\xi_1 \\
\cdots&
\end{aligned}$$

$$\begin{aligned}
\mathrm{d}\mathbf{v}_i^{\mathcal{M}} =& \mathrm{d}\mathbf{v}_0^{\mathcal{M}} \cdot \prod_{j=0}^{i-1} D_1 f(\mathbf{v}_j^{\mathcal{M}}, \xi_j) \\
& + \sum_{k=0}^{i-2} \mathrm{d}\xi_k \cdot D_2 f(\mathbf{v}_k^{\mathcal{M}}, \xi_k) \prod_{j=k+1}^{i-1} D_1 f(\mathbf{v}_j^{\mathcal{M}}, \xi_j) \\
& + \mathrm{d}\xi_{i-1} \cdot D_2 f(\mathbf{v}_{i-1}^{\mathcal{M}}, \xi_{i-1}).
\end{aligned} \tag{9}$$

As illustrated in Fig. 6, $\mathrm{d}\mathbf{v}_k^{\mathcal{M}} \sim \mathrm{d}\mathbf{v}_{k+1}^{\mathcal{M}}$ ($k = 0, \dots, i-1$) are equivalent infinitesimals at arbitrary points of surfaces (assuming that the material is locally consistent, and thus $O(D_1 f) = O(1)$), and $D_2 f > 0$ for non-Dirac surfaces. So, the infinitesimal change of a manifold vertex $\mathrm{d}\mathbf{v}_i^{\mathcal{M}}$ is equivalent to

$$\mathrm{d}\mathbf{v}_i^{\mathcal{M}} \sim \mathrm{d}\mathbf{v}_0^{\mathcal{M}} + \sum_{k=0}^{i-1} \mathrm{d}\xi_k \cdot D_2 f(\mathbf{v}_k^{\mathcal{M}}, \xi_k), \tag{10}$$

indicating that it is accumulated by the change of initial position (e.g., that due to pixel jittering or aperture samples) and the changes of quasi-random vectors at non-Dirac vertices. Theorem 1 is thus proven, supporting our inference in Sec. 3.1.
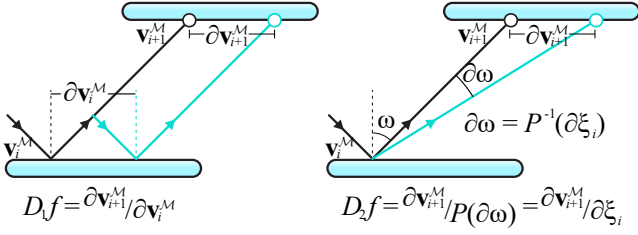
**Figure 6:** *Intuitive visualization of the partial derivatives of function $f$. $P$ is the cumulative distribution function (CDF) of the density function $p$.*
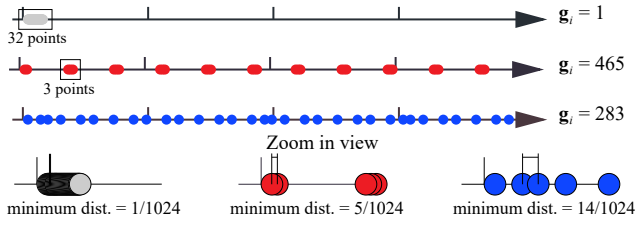


**Figure 7:** *The top axis: the first 32 points of different candidate components clumps at the beginning of the axis. The middle axis: every 3 points clumps at a minimum distance of $5/1024$. The bottom one is with larger minimum distance distance of $14/1024$ close to a uniform distribution. (all $n = 1024$)*

#### 3.2.3. Discussion

To summarize, our novel framework, in contrast to traditional approaches, shall be built upon permuting components of the existing generator vectors. Given an initial generator vector $\mathbf{g}$, we find a local optimal permutation that maximizes the minimum distance of each 2-D pair of dimensions for arbitrary high dimensionality. As an advantage over previous algorithms, the proposed algorithm tends to terminate in a much more reasonable time. However, there is still room for further optimization on higher-dimensional pairings. The implementation details of the framework will be described in later sections, where the optimization will be extended into higher dimensionality to ensure more robustness for the lattices.

### 3.3. Preliminary components: on a lower bound of minimum distance

So far, our primary task is to determine the initial components for the generator vector $\mathbf{g}$. Any component $\mathbf{g}_i$ that fulfills $\gcd(\mathbf{g}_i, n) = 1$ fills the identical intervals on the coordinate axis without having overlapping points. The difference of these components, as shown in Fig. 7, presents in the incomplete sequence: some distribute more evenly on the axis, while others form a cluster in part of the axis, which will lead to the undesirable convergence curve in Fig. 2.

Here we prove how the unevenly distributed samples on axes

influence the incomplete lattice. Assume $\mathbf{x}_i$ and $\mathbf{x}_j$ being a pair of lattice points in the $s$-dimensional point set $P_n$ with $\|\mathbf{x}_i - \mathbf{x}_j\|$ being the minimum distance

$$d_{min}(P_n) = \sqrt{\sum_{k=1}^{s}\left(\mathbf{x}_{i,k} - \mathbf{x}_{j,k}\right)^2} \geq \frac{\sum_{k=1}^{s}\left|\mathbf{x}_{i,k} - \mathbf{x}_{j,k}\right|}{\sqrt{s}}$$
$$\geq \frac{\sum_{k=1}^{s}\min_{0 \leq a < b < n}\left|\mathbf{x}_{a,k} - \mathbf{x}_{b,k}\right|}{\sqrt{s}} \quad (11)$$

where $\mathbf{x}_{i,k}$ denotes the $k^{\text{th}}$ component of vector $\mathbf{x}_i$. Eq. 11 gives a lower bound of the minimum distance for the lattices, indicating that the per-axis distribution has an impact on the overall quality. In response, the proposal of selecting the preliminary components in our framework is based on maximizing this bound.

We consider all possible lengths of an incomplete sequence, using a rank

$$r(\mathbf{g}_k) = \left(\prod_{m=2}^{n} m \cdot \min_{0 \leq i < j < m}\left|\mathbf{x}_{i,k} - \mathbf{x}_{j,k}\right|\right)^{1/(n-1)} \quad (12)$$

to depict the average evenness of points according to the component $\mathbf{g}_k$. Theoretically, the vector should then be initialized with the first $s$ candidate components of the highest rank.

## 4. Our method: technical presentation

As outlined in the blueprint, our framework consists of two steps, namely: selecting the preliminary components, and finding an optimal permutation based on the pairwise optimization. In this section, we provide the concrete technical details. First, in Sec. 4.1, we make three assumptions of the parameters for our rank-1 lattice. Then, in Sec. 4.2, we tackle the challenge of the high computational expense due to looking for the mutual minimum distance for incomplete sequence by using a fast estimation approach. Finally, in Sec. 4.3, we complete the framework by introducing the high-dimensional extension of the 2-D pairwise permutation.

### 4.1. Assumptions on the lattice parameters

Since we rely on several fast approaches, we make three assumptions regarding the lattice parameters to facilitate the follow-up discussion.

**Assumption 1: The total number of lattice points $n$ is power of 2.** Two benefits can be drawn from this: the first is that a floating point number divided by a power of 2 is generally fast. The second is more robustness of the computation. For example consider a large $n$ hitting the maximum integer, and only when $n$ being a power of 2 can it be guaranteed that

$$\mathbf{x}_i = \frac{\left(i\mathbf{g} \bmod 2^b\right) \bmod n}{n} \equiv \frac{i\mathbf{g} \bmod n}{n}, \quad (13)$$

where $2^b - 1$ is the maximum unsigned integer in binary computers.

**Assumption 2: The dimension of generator vector g is power of 2.** This is required by the swapping strategy extending into a higher dimensionality in Sec. 4.3.

**Assumption 3: The components of g are odd.** With the previous

two assumptions, all components being odd integers efficiently satisfies $\gcd(\mathbf{g}_i, n) = 1$, such that the points fill every integer interval without overlapping.

## 4.2. Fast realization of mutual minimum distance for incomplete sequences

In Eq. 12, we present a rank that theoretically measures the average evenness of points on each axis. However, the fact is that performing a mutual search and accounting for all possible length are both conducive to a complexity of $O(n^3)$ as well as possessing little achievability. In addition, the criterion for pairwise optimization is based on the same idea, where the computation will be more violent. We seek a faster realization of the criterion.

**Simplifying the mutual search.** The naive exhaustive search for the mutual minimum distance of a full lattice is identical to searching the toroidal minimum distance to the origin of a non-zero value [Dam09] due to the regular structure

$$d_{min}(\mathcal{L}_n) = \min_{0 \leq i < j < n} \|\mathbf{x}_i - \mathbf{x}_j\| = \min_{1 \leq i < n} \|\mathbf{x}_i - \mathbf{x}_0\| . \quad (14)$$

When the sequence is incomplete, the second equal sign in Eq. 14 does not establish, since the incomplete sequence of points does not have a regular structure, and the minimal distance to the origin may not equal to that to the other points. However, this distance is always slightly greater (or equal) to the mutual minimal distance of all points

$$\min_{0 \leq i < j < n} \|\mathbf{x}_i - \mathbf{x}_j\| \leq \min_{1 \leq i < n} \|\mathbf{x}_i - \mathbf{x}_0\| = \min_{1 \leq i < n} \sqrt{\sum_{k=1}^{s} \min\left(\mathbf{x}_{i,k}, 1 - \mathbf{x}_{i,k}\right)^2},$$

$$(15)$$

which is an upper bound and fast estimation of the mutual minimal distance.

**Reusing existing information.** As a side effect, Eq. 15 does not account for the mutuality, such that computing the estimation for $n$ points requires simply comparing the length of the $n^{\text{th}}$ vector to the current minimum value of previous $n - 1$ vectors. In this context, the computational cost in Eq. 12 is reduced to $O(n)$.

**Proposition.** In practice, using the above simplification technique, the preliminary components produced by Eq. 12 are often not perceived as the candidates of the best quality of average distribution. This is because shorter sequences display more evident distribution defects. Longer sequences should be less weighted. We opt for making the weighting similar to the Sobol′ and van der Corput sequences that are optimal at the length of the sequence being to the power of 2 [OG18]

$$r'(\mathbf{g}_a, \mathbf{g}_b, \dots) =$$

$$\left( \prod_{m=1}^{\log_2 n} 2^{m/l} \cdot \min_{1 \leq i < 2^m} \sqrt{\sum_{k=1}^{s} \min\left(\mathbf{x}_{i,k}, 1 - \mathbf{x}_{i,k}\right)^2} \right)^{1/(\log_2 n - 1)},$$

$$a, b, \dots \in \{1, \dots, s\} \quad (16)$$

where $\mathbf{x}$ is the lattice point set created from generator vector $\mathbf{g} = (\mathbf{g}_a, \mathbf{g}_b, \dots)$ and $l$ is the argument length of $r'$.

**Initializing g.** Given all possible candidates components $\mathbf{c} =$

$(\mathbf{c}_1, \dots, \mathbf{c}_{n/2}) = (1, 3, \dots, n-1)$, $\mathbf{g}$ is initialized to the largest $s$ components of $\mathbf{c}$ sorted descendingly by rank $r'$, as shown in Alg. 1

---

**Algorithm 1** Initialize **g**

---

1: $\mathbf{c} \leftarrow (1, 3, \cdots, n-1)$      ▷ Initialize all candidates
2: $\mathbf{sort}(\mathbf{c}, r'(\mathbf{c}_i) > r'(\mathbf{c}_j))$      ▷ Sort by rank $r'$
3: **for** $i \leftarrow 1$ to $s$ **do**
4:      $\mathbf{g}_i \leftarrow \mathbf{c}_i$      ▷ Retrieve the first $s$ candidates
5: **end for**

---

## 4.3. The swapping strategy with the higher-dimensional optimization

**The two-dimensional example.** First we provide the example of the 2-D case, which assists the readers to understand the more complicated multidimensional cases. The basic idea is that, for a given set of components $\mathbf{g} = (g_1, \dots, g_{2^k})$ initialized using criterion $r'$, we repeatedly swap $\mathbf{g}_{2i}$ with $\mathbf{g}_j$ where $j \in \{2i+1, \dots, 2^k\}$ and $r'(\mathbf{g}_{2i-1}, \mathbf{g}_j)$ has the maximum value for $i = 1, \dots, 2^{k-1}$. The pairing of every two components is thus locally optimized via the greedy approach.

**Multidimensional cases.** The higher-dimensional swapping strategy can be generalized from the 2-D example with the following steps:

1. $r'(\mathbf{g}_{2i-1}, \mathbf{g}_{2i})$ for $i = 1, \dots, 2^{k-1}$ is repeatedly maximized using the scheme as discussed above;
2. Every 2-D subset in step 1 is regarded as a whole (preserving the internal order), and the same scheme to maximize $r'(\mathbf{g}_{4i-3}, \mathbf{g}_{4i-2}, \mathbf{g}_{4i-1}, \mathbf{g}_{4i})$ for $i = 1, \dots, 2^{k-2}$ is used;

⋮

k-2. Every $(k-2)$-D subset in step $k-3$ is regarded as a whole to maximize $r'(\mathbf{g}_{2^{k-2}(i-1)+1}, \dots, \mathbf{g}_{2^{k-2}})$ for $i = 1, 2$.

The algorithm is interpreted in Fig. 8. The complexity of the whole framework is $O(ns^2)$.
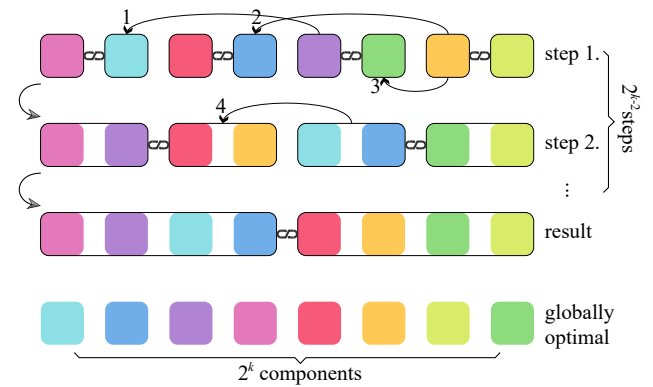


**Figure 8:** *An intuitive interpretation of the swapping strategy. The color blocks indicate the preliminary components to permute. The numbers labelled on the arrows are the order of swapping.*

## 5. Results

In this section, we perform a series of experiments to verify the effectiveness of our vector search framework and the performance of our rank-1 lattices.

**Offline search cost for the generator vector.** We present the measured computational time for searching the generator vector by comparing our framework with the CBC algorithm. As expected, in Fig. 9, given the total number of points $n = 4096$, the computational time increased quadratically with dimensionality $s$ for our method. The fixed cost of approximately 0.4 sec (which is evident at $s \leq 16$) was due to sorting of the preliminary candidates. In contrast to this, the CBC algorithm failed to terminate in an acceptable time when $s > 16$.

**Variances in different stages.** In Fig. 10, we demonstrate the variance convergence corresponding to each stage in our framework rendering the *Cornell Box* scene, namely before pairwise swapping (labelled "Preliminary"), after performing the 2-D optimization as discussed in Sec. 3.2.1 (labelled "2-D Opt."), and completing the full framework (labelled "Full Framework"). Again, as expected, the 2-D optimization was conducive to a major improvement in quality, and the full optimization precipitated a coherent convergence curve even superior to the Sobol$'$ sequence.

**On-line computational speed.** In Fig. 11, we compared the computational cost of our lattice to several samplers. We set the sample count to simulate a sensor of a $1920 \times 1080$ resolution rendered at 64 SPP using 16-D quasi-random vectors. Because of the simple congruential computation, the lattice demonstrated a distinct ascendancy over the compared sequences, which will benefit the overall render time.

**Render performance.** We evaluated the real-time render performance in game engine *Unreal Engine 4* and the offline performance in *PBRT* [PJH17a].

The real-time performance was manifested in Fig. 12. We compared the per-sample render time of our lattice with the Sobol$'$ [Sob67] and Halton [Hal64,ÖSG12] sequences. The numerical performance differences are labelled with the relative time proportional to our lattice as percentages. We demonstrate two cases, which are rendering with and without the *Bounding Volume Hierarchy* (BVH) acceleration structures [Wal07]. In the first two rows, we tested two scenes with traditionally defined triangle meshes and BVH acceleration structures. Our lattice achieved a $2\% \sim 7\%$ performance boost. The lower two rows presented the performance in simpler scenes (and thus without the use of BVH) where the ray traversed faster and the computational cost mostly fell onto computing the quasi-random sequences. In these scenes, our lattice was up to multiple times faster than the LDS.

The performance in offline rendering is shown in Fig. 17. We measured the variance in 3 *PBRT*'s classic scenes using the Sobol$'$ sequence and our rank-1 lattice. The statistics of average performance (measured by $\frac{\text{Sobol}'}{\text{ours}} \times 100\%$) are as follows:

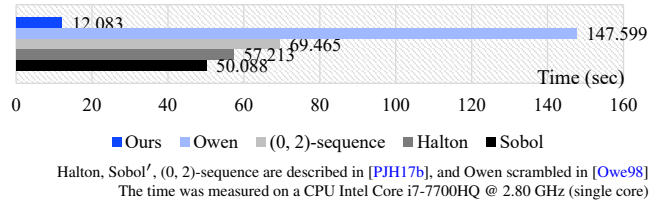| scene | avg. path length | avg. perf. (full) | avg. perf. (crop) |
|---|---|---|---|
| *Volkswagen Van* | 1.434 | 137.66% | 230.19% |
| *Imperial Crown* | 4.136 | 96.00% | 107.62% |
| *Sportscar* | 1.502 | 101.29% | 116.33% |
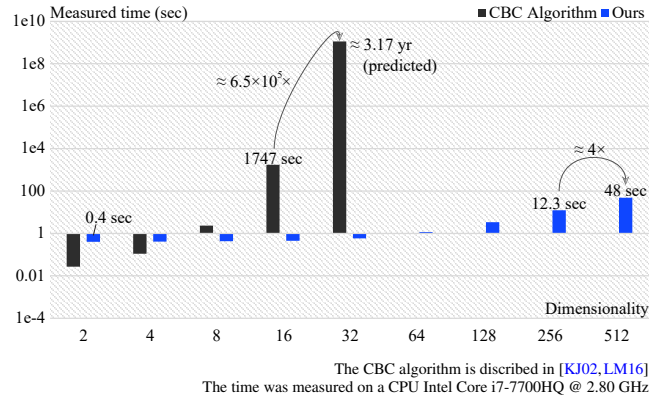


**Figure 9:** *Measured computational time for generating lattices of $n = 4096$.*
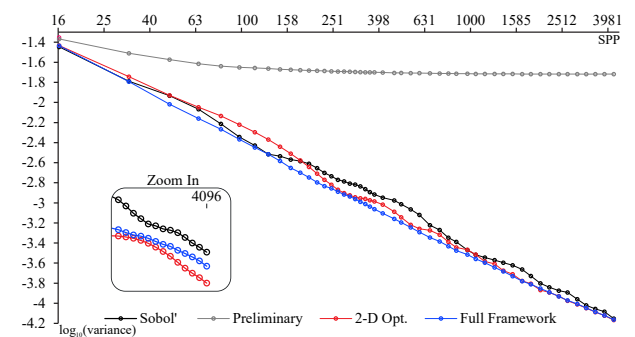


**Figure 10:** *Variance in each stage of our framework, compared to random sampling and the Sobol$'$ sequence.*



**Figure 11:** *Cost of generating $1920 \times 1080 \times 64$ 16-D samples. Although the (0, 2)-sequence is known for its high efficiency in 2-D, the high-dimensional shuffling leads to an extra unneglectable expense.*

As shown, our rank-1 lattice achieves an evident performance improvement (especially at shorter path lengths), while in the worst case producing no significant drawbacks.

## 6. Discussion and conclusion

In this paper, we have demonstrated a proof-of-concept of integration lattices in solving path integrals and an efficient lattice search algorithm which, by initializing preliminary components and performing pairwise permutation, achieves arbitrary sample counts
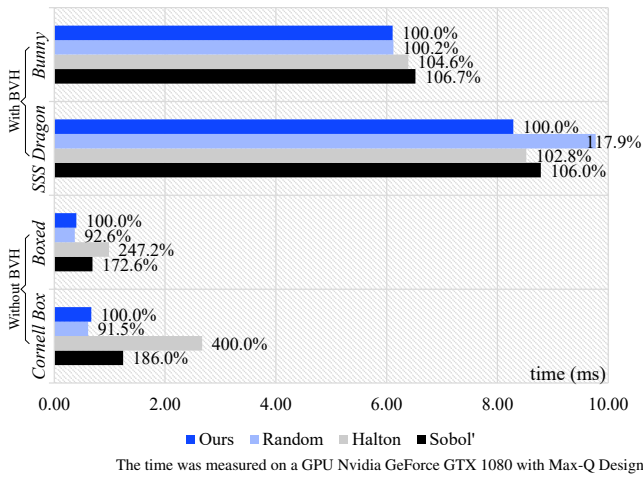
**Figure 12:** *Average per-sample elapsed time at* $512 \times 512$ *resolution in real-time rendering. The data were collected through the average of 25600 samples.*
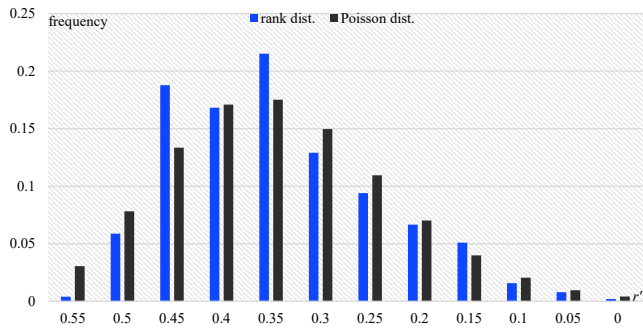


**Figure 13:** *The distribution of the ranks fits a Poisson distribution very well, with* $n = 1024$.

and dimensionality. In addition, we investigated the rippling effect in path integration, and proposed that 2-D optimization is the key to better convergence of the renderings. In this section, we discuss several topics of interest and limitations of the present work.

**Subject 1: The optimal number of dimension.** In our research, one of the interesting discoveries was that the criterion ranks $r'$ of all possible candidate components distribute as a Poisson distribution, as shown in Fig. 13. If the dimensionality is not specified, the instinctive choice of the optimal number of dimension is the expected value $\lambda$ of the fitted Poisson distribution. Reminiscent of **Assumption 2** in Sec. 4.1 that the dimensionality is a power of 2, we propose using the 25% best candidates, which is approximately the expected value.

**Subject 2: Constructing per-pixel unique sequences.** Unlike the Halton sequence through radical inversions and the Sobol′ sequence through binary matrices, there is no analytical method to inverse a sampling interval (pixel) to the sample index to generate per-pixel unique sequence. For example, the "Original" lattice in
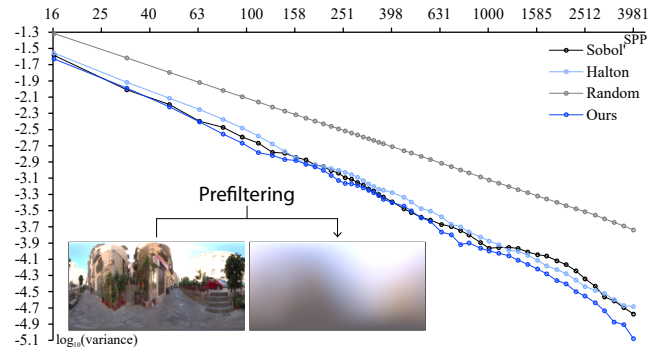


**Figure 14:** *Variance in cosine-filtering an environment cubemap.*

Fig. 5 does not even fill all pixels, and the inversion is thus invalid. We opt for using *Cranley-Patterson Rotations* [KK02] that use a random mask to decorrelate the samples between pixels. We adopt the trending *Blue-noise Dithered Sampling* (BNDS) [GF16] technique that achieves better visual fidelity at low sample counts, addressing the aliasing effect in the Halton and Sobol′ samplers. Since the rank-1 lattices themselves are in recurrence form, *Cranley-Patterson Rotations* have no influence on the convergence of the estimation.
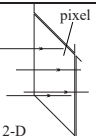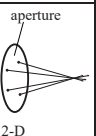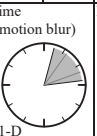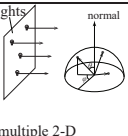
**Subject 3: MMD lattices for non-Lipschitz continuous functions.** MMD lattices are very effective in estimating the integration of Lipschitz continuous functions [KN12,Dam09] . However, many scholars argue that their effectiveness for non-Lipschitz continuous functions (especially in high dimensionality) is yet to be validated. Specifically, path integration is usually discontinuous, e.g., on the edge of a geometry. When we recursively optimize the lattices, we eventually obtain MMD-like lattices. Fortunately, their effectiveness was validated by our experiment results.

**Bonus: better sampling efficiency for 2-D integrals.** The by-product of our 2-D optimization is demonstrated in Fig. 14, by examining the variance of cubemap prefiltering, which is commonly used for the light probes in real-time game engines [Gre03,Hoo16]. Thanks to the optimized minimum distance, our lattice ($n = 4096$) achieves significant advantages over other compared sequences.
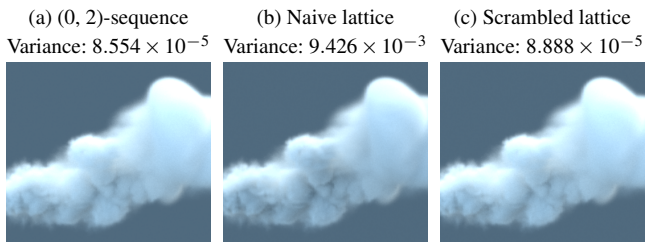
**Limitation: arbitrary pairing of dimensions.** Our lattice does not support random pairing of dimensions. Since our search algorithm is based on permutation, making use of the rippling effect to optimize the lattices for path integration, randomly choosing the dimensions for sampling will result in the lattices falling back to conventional ones, producing unstable convergence curve as in Fig. 2 or an even worse distribution as the "Original" lattice in Fig. 5.

**Limitation: odd-dimensional elements in the path integral.** In addition to the previous limitation, our lattice requires skipping one dimension at each odd-dimensional element (e.g. time for motion blur and *Russian roulette*) as shown in Fig. 15. This is because the odd-dimensional elements prevent the later process sampling from the optimized 2-D quasi-random vectors.

**Limitation: volumetric light transport.** Throughout the paper our core idea relies on optimizing the pairing of dimensions to achieve

| Components | $\mathbf{x}_{i,1}$ | $\mathbf{x}_{i,2}$ | $\mathbf{x}_{i,3}$ | $\mathbf{x}_{i,4}$ | $\mathbf{x}_{i,5}$ | $\mathbf{x}_{i,6}$ | $\mathbf{x}_{i,7}$ | $\mathbf{x}_{i,8}$ | … |
|---|---|---|---|---|---|---|---|---|---|
| Other samplers | pixel | | aperture | | time | lights, BRDF, … | | | |
| Our sampler | pixel | | aperture | | time | - | lights, BRDF, … | | |
| Example |  | | | | | | | | |

**Figure 15:** *Allocation of the quasi-random components. Our sampler deliberately skips one quasi-random component at the odd-dimensional time component.*

(a) (0, 2)-sequence
Variance: $8.554 \times 10^{-5}$

(b) Naive lattice
Variance: $9.426 \times 10^{-3}$

(c) Scrambled lattice
Variance: $8.888 \times 10^{-5}$



**Figure 16:** *Random Digit Scrambling resolves the limitation of our lattice in rendering volumetric light transport.*

better path vertex distribution in the 2-D manifold space for regular meshes. However, it is violated by the volumetric scattering effect, whose virtual intersections are in 3-D manifold space. In Fig. 16(b), we show that our naive lattice failed to produce an unbiased result. To address this, we use *Random Digit Scrambling* [KK02] to scramble the sample index $i$ with a random scrambling vector $\mathbf{u}$ whose components are identical at each two dimensions

$$\mathbf{x}_i = \frac{(i \oplus \mathbf{u}) \cdot \mathbf{g} \bmod n}{n}, i = 0, \dots, n-1 \quad (17)$$

where $\oplus$ is the binary exclusive-or operation. The result using the scrambled sequence, as shown in Fig. 16(c), achieves unbiased convergence similar to *PBRT*'s (0, 2)-sequence.

**Future work.** Despite the promising results achieved by the current stage of our work, we sketch a road map of future work.

**Other lattice-search strategies.** To utilize the optimized rank-1 lattice in path integral estimation, we initialize the preliminary components by maximizing a lower bound of minimum distance. However, artifacts exist in arbitrary pairing of dimensions among the preliminary components. A more robust approach to initialize them shall solve this problem. Also, the way of pairwise optimization is not unique, and there is possibly a better solution than the greedy approach presented in this paper.

**Random Digit Scrambling.** In Fig. 16, we have seen a glimpse of how *Random Digit Scrambling* cooperates with rank-1 lattices, which introduces only one exclusive-or operation, preserving the high efficiency of the computation. It yields a new possible solution for constructing dimensionally robust integration lattices with

optimal 2-D projections by searching a sequence of optimal scrambling keys.

**The rippling effect: a possible solution for future design of sampling sequences.** As one of our core contributions, the rippling effect is introduced to the construction of sampling lattices. We look forward to further investigations in future work to design the state-of-the-art samplers with more sampling efficiency in path integral estimation.

## 7. Acknowledgement

## References

[Bre11] BREMNER M.: *Lattice basis reduction: An introduction to the LLL algorithm and its applications.* CRC Press, Aug. 2011. doi:10.1201/b11066. 3

[Car02] CARY M.: Lattice basis reduction, algorithms and applications, Feb. 2002. URL: https://courses.cs.washington.edu/courses/cse590z/03au/cary.pdf. 2

[CKK18] CHRISTENSEN P., KENSLER A., KILPATRICK C.: Progressive multi-jittered sample sequences. *Computer Graphics Forum 37*, 4 (2018), 21–33. doi:10.1111/cgf.13472. 4

[Dam09] DAMMERTZ S.: *Rank-1 Lattices in Computer Graphics.* PhD thesis, Institut für Medieninformatik, Ravensburg, Germany, 2009. doi:10.18725/OPARU-1068. 2, 3, 4, 7, 9

[DKD08] DAMMERTZ H., KELLER A., DAMMERTZ S.: Simulation on rank-1 lattices. *Monte Carlo and Quasi-Monte Carlo Methods 2006* (2008), 205–216. doi:10.1007/978-3-540-74496-2_11. 2

[FDL14] FINCKH M., DAMMERTZ H., LENSCH H.: On near optimal lattice quantization of multi-dimensional data points. *Computer Graphics Forum 33*, 1 (2014), 271–281. doi:10.1111/cgf.12273. 4

[GF16] GEORGIEV I., FAJARDO M.: Blue-noise dithered sampling. pp. 1–1. doi:10.1145/2897839.2927430. 9

[Gre03] GREEN R.: Spherical harmonic lighting: The gritty details. 9

[Hal64] HALTON J. H.: Algorithm 247: Radical-inverse quasi-random point sequence. *Commun. ACM 7*, 12 (Dec. 1964), 701–702. doi:10.1145/355588.365104. 8

[Hel85] HELFRICH B.: Algorithms to construct minkowski reduced and hermite reduced lattice bases. *Theoretical Computer Science 41* (1985), 125–139. doi:10.1016/0304-3975(85)90067-2. 2

[Hoo16] HOOKER J.: Volumetric global illumination at treyarch, Aug. 2016. URL: https://research.activision.com/publications/archives/volumetric-global-illumination-at-treyarch. 9

[JSM08] JALDEN J., SEETHALER D., MATZ G.: Worst- and average-case complexity of lll lattice reduction in mimo wireless systems. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing* (2008), pp. 2685–2688. doi:10.1109/ICASSP.2008.4518202. 4
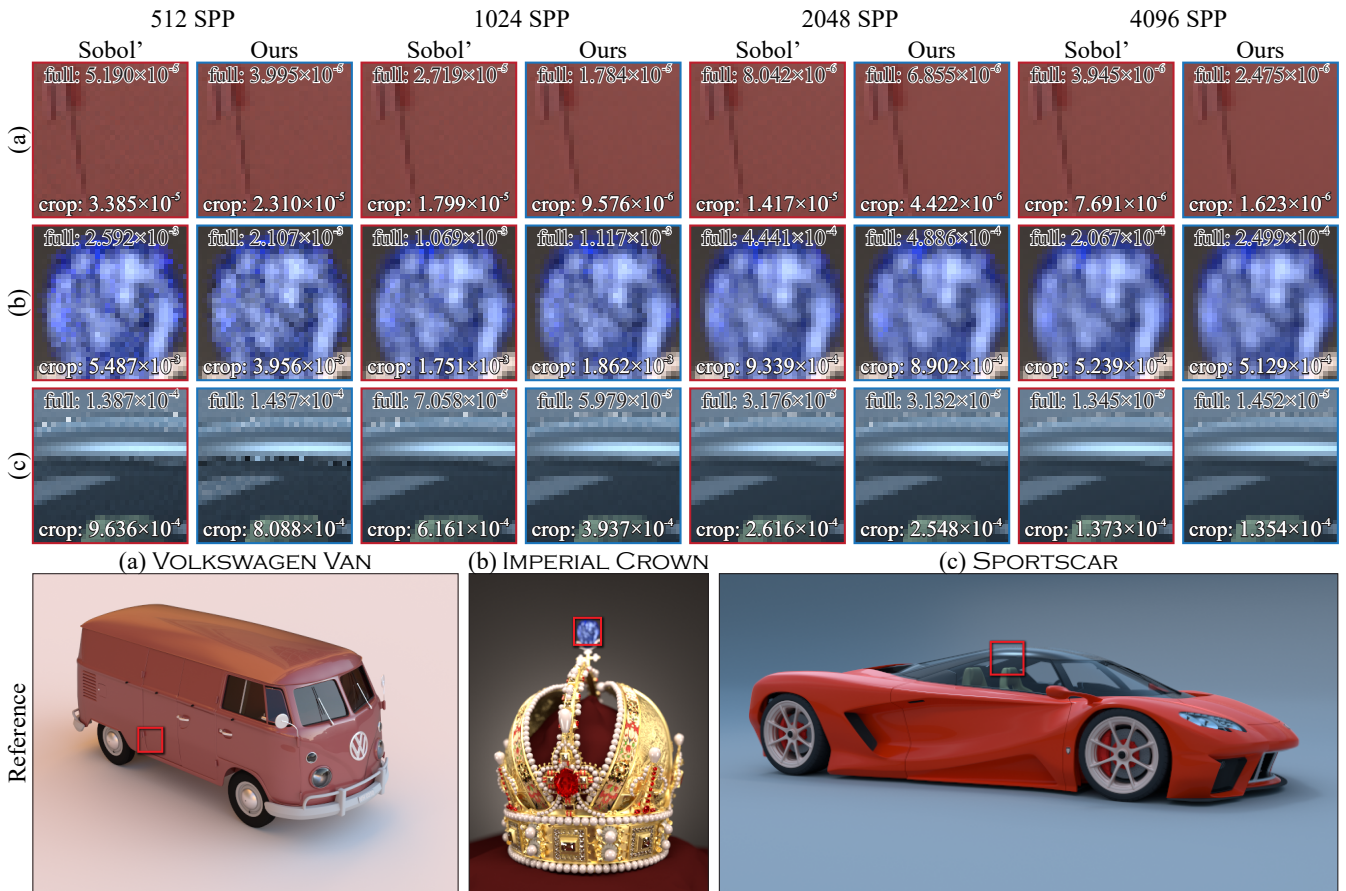
**Figure 17:** *Renderings of Volkswagen Van, Imperial Crown, and Sportscar. The 3 classic scenes feature different combinations of DoF, soft shadows, and microfacet models. Our rank-1 lattice is compared to the Sobol′ sequence at different samples per pixel (SPP). Our rank-1 lattice achieves convergence close to the Sobol′ sequence.*

[KJ02] KUO F. Y., JOE S.: Component-by-component construction of good lattice rules with a composite number of points. *Journal of Complexity 18*, 4 (2002), 943 – 976. `doi:10.1006/jcom.2002.0650`. 2, 4, 8

[KK02] KOLLIG T., KELLER A.: Efficient multidimensional sampling. *Computer Graphics Forum 21*, 3 (2002), 557–563. `doi:10.1111/1467-8659.00706`. 2, 9, 10

[KN12] KUIPERS L., NIEDERREITER H.: *Uniform Distribution of Sequences*. Dover Books on Mathematics. Dover Publications, 2012. `doi:10.1137/1019028`. 9

[Llo82] LLOYD E. K.: The art of computer programming, vol. 2. *Software: Practice and Experience 12*, 9 (1982), 883–884. `doi:10.1002/spe.4380120909`. 3

[LM16] L'ECUYER P., MUNGER D.: Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. *ACM Trans. Math. Softw. 42*, 2 (May 2016). `doi:10.1145/2754929`. 2, 3, 8

[MBR*13] MARQUES R., BOUVILLE C., RIBARDIÈRE M., SANTOS L. P., BOUATOUCH K.: Spherical fibonacci point sets for illumination integrals. *Computer Graphics Forum 32*, 8 (2013), 134–143. `doi:10.1111/cgf.12190`. 2

[Nie87] NIEDERREITER H.: Point sets and sequences with small discrepancy. *Monatshefte für Mathematik 104*, 4 (1987), 273–337. `doi:10.1007/bf01294651`. 2, 4

[OG18] OWEN A. B., GLYNN P. W.: *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC 2016, Stanford, CA, August 14-19*. Springer International Publishing, 2018. `doi:10.1007/978-3-319-91436-7`. 2, 3, 7

[ÖSG12] ÖKTEN G., SHAH M., GONCHAROV Y.: Random and deterministic digit permutations of the halton sequence. `doi:10.1007/978-3-642-27440-4_35`. 8

[Owe98] OWEN A. B.: Scrambling sobol' and niederreiter–xing points. *Journal of Complexity 14*, 4 (1998), 466 – 489. `doi:10.1006/jcom.1998.0487`. 8

[PJH17a] PHARR M., JAKOB W., HUMPHREYS G.: pbrt-v3, 2017. URL: `https://github.com/mmp/pbrt-v3/`. 8

[PJH17b] PHARR M., JAKOB W., HUMPHREYS G.: *Physically Based Rendering: From Theory to Implementation*, 3rd ed. Morgan Kaufmann Publishers/Elsevier, 2017. `doi:10.1016/C2013-0-15557-2`. 4, 8

[Sob67] SOBOL' I. M.: On the distribution of points in a cube and the approximate evaluation of integrals. *U.S.S.R. Comput. Math. Math. Phys. 7* (1967), 86–112. `doi:10.1016/0041-5553(67)90144-9`. 2, 8

[Thi01] THIÉMARD E.: An algorithm to compute bounds for the star

discrepancy. *Journal of Complexity 17*, 4 (2001), 850 – 880. `doi:10.1006/jcom.2001.0600`. 4

[Vea98]  VEACH E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Stanford, CA, USA, 1998. AAI9837162. 1, 3

[Wal07]  WALD I.: On fast construction of sah-based bounding volume hierarchies. In *Proceedings of the 2007 IEEE Symposium on Interactive Ray Tracing* (USA, 2007), RT '07, IEEE Computer Society, p. 33–40. `doi:10.1109/RT.2007.4342588`. 8