Volume 33 (2014), Number 1 pp. 271–280 COMPUTER GRAPHICS *forum*

On Near Optimal Lattice Quantization of Multi-Dimensional Data Points

M. Finckh[1] and H. Dammertz[2] and H. P. A. Lensch[3]

[1] manuel.finckh@gmail.com Realtime Technology AG
[2] holger.dammertz@gmail.com Realtime Technology AG
[3] hendrik.lensch@uni-tuebingen.de Tübingen University

Abstract

One of the most elementary application of a lattice is the quantization of real valued s-dimensional vectors into finite bit precision to make them representable by a digital computer. Most often, the simple s-dimensional regular grid is used for this task where each component of the vector is quantized individually. However, it is known that other lattices perform better regarding the average quantization error. A rank-1 lattices is a special type of lattice, where the lattice points can be described by a single s-dimensional generator vector. Further, the number of points inside the unit cube $[0,1)^s$ is arbitrary and can be directly enumerated by a single one-dimensional integer value. By choosing a suitable generator vector the minimum distance between the lattice points can be maximized which, as we show, leads to a nearly optimal mean quantization error. We present methods for finding parameters for s-dimensional maximized minimum distance rank-1 lattices and further show their practical use in computer graphics applications.

Categories and Subject Descriptors (according to ACM CCS): I.4.1 [Computer Graphics]: Digitization and Image Capture—Quantization

1. Introduction

The use of quantization of data points is one of the most basic and often used tools for algorithms in computer graphics. For example colors are quantized in 8-bit per color channel for image display or normal vectors are packed into 24-bit normal maps. This principle is easily generalized to arbitrary dimensions s by quantizing each dimension separately. This can then be used in practical applications for almost any data that occurs in a graphics algorithm.

More formally, this kind of quantization is the construction of a s-dimensional regular grid where the number of discrete points is 2^{bs} (with b being the number of bits chosen to represent a value per dimension). While for specific use-cases there are always highly sophisticated methods for data reduction or quantization that lead to superior results, the regular grid is still widely used. It is easy to implement and understand; quantization is straightforward and fast to convert back to the original domain. It is thus an indispensable tool for the computer graphics practitioner.

Despite its wide use the regular grid has two major drawbacks. First the number of discretization points is the product of the number of points per dimension (i.e. for 8-bit RGB images it is 256^3) and it is not possible to arbitrarily adjust the number of points and thus the bit depth to, for example, 17. Adjusting the bit-depth per dimension has the problem of different quantization errors per dimension and thus is not optimal, if, for example all dimensions are equally important. The second drawback is that it is known that the regular grid is not optimal regarding the quantization error for $s > 1$ (for $s = 1$ the regular grid is just equidistant points).

In this paper we propose and discuss the use of a replacement for the regular grid quantization by the use of rank-1 lattices. These lattices solve both problems mentioned above while still being as simple and fast to use in practical applications. Even though the mathematical theory of these lattices can be quite complex, we show in this paper that they can be easily used for quantization in any algorithm. We start in Section 3 by giving a more formal introduction to general lattice quantizers in s dimensions and define the quantization error. In Section 4 we present the theory of rank-1 lattices and show how they can be constructed to perform always better than the regular grid.

Section 5 gives a step-by-step practical guide on how to use our proposed quantization scheme in combination with a table containing commonly used bit-depths up to 6 dimensions.

2. Background

The scope of our work is to present a general quantization approach using rank-1 lattices that can be used as a replacement for ad-hoc regular grid quantization. We thus target no specific use case besides representing multi-dimensional data with a fixed pre-allocated amount of depth. Here we concentrate on presenting the related work regarding the theoretical foundation of the use of lattices. For any specific use case there are many possible choices of compression algorithms that perform better than any general quantization.

As a generalization of lattice quantization, vector quantization is extensively studied for many application domains. See for example [GG91]. Compared to our approach it is a lot more complex to implement and use however.

Lattices other than the s-dimensional Cartesian lattice $\mathbb{Z}^s$ have been studied for their use in computer graphics applications. The hexagonal lattice has been used because of its optimal sampling efficiency in two dimensional space [VVPL02, MS05, CVFH08]. In particular the problem of re-sampling between regular and hexagonal images is addressed. The permutohedral lattice has been used for high dimensional data filtering [ABD10].

The general theory of s-dimensional optimal lattices is connected to sphere packings and their applications, where [CS10] provides a comprehensive discussion. Because of its optimal sampling efficiency in three dimensional space, the *body centered cubic* lattice (BCC) has been used for sampling and visualization of volumetric data [EVDVM08, Cse05, LLWQ13]. In addition, the 4D BCC lattice is used for 4D data visualization in [NM02, LQ11].

Directly related to our work are applications of two dimensional rank-1 lattices in computer graphics. They have been studied extensively by Dammertz et al. for image synthesis and texture processing in [DK08, DDKL09]. Algorithms for finding parameters for two-dimensional rank-1 lattices in order to maximize the minimum distance between the lattice points are presented in [DDK08, Dam09]. These *maximized minimum distance* (MMD) rank-1 lattices closely approximate the hexagonal lattice. Notably in [DDKL09] the equivalence of hexagonal lattices and a special class of rank-1 lattices is proven.

3. Lattice Quantizers

In this Section we first give a brief introduction to lattices and quantizers in general. Then we discuss the properties of lattice quantizers. For a more detailed and rigorous treatment of the subject we refer the reader to [CS10].

3.1. Lattices

A lattice is a discrete additive subgroup of $\mathbb{R}^s$, i.e. a subset $\Lambda \subseteq \mathbb{R}^s$ which satisfies:

subgroup Λ is closed under addition and subtraction,

discrete there is an $\varepsilon > 0$ such that two distinct points $x \neq y \in \Lambda$ are at distance at least $\| x - y \| \geq \varepsilon$.

The second rule is only important for theoretical considerations as this rule is enforced by the finite precision of a digital computer. The first rule, however, ensures that the basic operations of addition and subtraction do not lead to additional quantization (round-off) errors. A simple example of a lattice is the s-dimensional cubic lattice $\mathbb{Z}^s$.

Lattice Basis: Each point of an s-dimensional lattice can be represented as an integer linear combination of s linear independent basis vectors $\mathbf{b_1}, ..., \mathbf{b_s}$:

$$\mathbf{x} = \sum_{i=1}^{s} \xi_i \mathbf{b_i} \,, \tag{1}$$

where $\xi = (\xi_1, ..., \xi_s)$ is an arbitrary integer vector. As such, the matrix $\mathbf{B} = (\mathbf{b_1}, ..., \mathbf{b_s})^T$ is also a *generator* of the lattice. Note that each matrix where any integer linear combination of the row vectors lead to the same points as in Equation (1) is a generator of the same lattice.

3.2. Quantizers

A general quantizer can be defined as follows:

- N points $P_1, ..., P_N \in \mathbb{R}^s$ are chosen.
- Input: arbitrary point $\mathbf{x} \in \mathbb{R}^s$.
- Output: closest P_i to $\mathbf{x}$.
- If closest P_i is not unique, chose one of them at random.

The procedure of quantization can also be described as:

- The points P_i partition the $\mathbb{R}^s$ into Voronoi cells $V(P_i)$,
- if $x \in V(P_i)$, the output is P_i.

Quantization Error: In general the *average mean squared error per dimension* for a given quantizer is

$$E = \frac{1}{s} \int_{\mathbb{R}^s} \| \mathbf{x} - P_i(\mathbf{x}) \|^2 \, p(x) \, dx \,, \tag{2}$$

where $P_i(\mathbf{x})$ is the closest point P_i to $\mathbf{x}$, $\| \cdot \|$ denotes the Euclidean norm, and $p(\mathbf{x})$ is the probability density of the input $\mathbf{x}$. As we are only concerned with lattice quantizers and we assume a uniformly distributed input, this can be simplified. The Voronoi cells $V(P_i)$ of a lattice are all congruent to each other and congruent to a polytope Π. If the origin of the coordinate frame is placed at the centroid of Π, (2) becomes

$$E = \frac{\frac{1}{s} \int_{\Pi} \mathbf{x} \cdot \mathbf{x} \, dx}{Vol(\Pi)} \,, \tag{3}$$

dim	Optimal Lattice	$G(\Pi)$	Regular Grid	$G(\Pi)$	Rank-1 $\mathbf{g} = (1, a, a^2, ...)$	$G(\Pi)$
1	$\mathbb{Z}$	0.083333	$\mathbb{Z}$	0.083333	$a = 1$	0.083333
2	A_2, hexagonal	0.080188	$\mathbb{Z}^2$	0.083333	$a = 25962$	0.080188
3	D_3^*, BCC	0.078543	$\mathbb{Z}^3$	0.083333	$a = 23128$	0.078752
4	D_4	0.076603	$\mathbb{Z}^4$	0.083333	$a = 20166$	0.076973
5	D_5^*	0.075625	$\mathbb{Z}^5$	0.083333	$a = 2616$	0.076320
6	E_6^*	0.074244	$\mathbb{Z}^6$	0.083333	$a = 20836$	0.075806

Table 1: *This table shows the quantization error $G(\Pi)$ from Equation 4 for the best known s-dimensional lattice quantizers (for an elaborate list and the construction of the higher-dimensional lattices D_4, D_5^* and E_6^* we refer to [CS10]), for the Cartesian lattice $\mathbb{Z}^s$, and for maximized minimum distance rank-1 lattices in Korobov form with 2^{16} points found through computer search. The value $G(\Pi)$ for the rank-1 lattices is computed via excessive Monte Carlo integration (10 million samples). It can be seen that the rank-1 lattices provide a near optimal quantization for each dimension.*

where $Vol(\Pi)$ is the volume of the Voronoi cell. This formulation is not independent of the scale of the lattice (i.e. expansion or contraction of the lattice with the same constant factor in each dimension would lead to a different error). In order to compare lattices of different scales, the error needs to be additionally normalized for the scale of the lattice:

$$G(\Pi) = \frac{E}{Vol(\Pi)^{\frac{2}{s}}} = \frac{\frac{1}{s}\int_\Pi \mathbf{x}\cdot\mathbf{x}\,dx}{Vol(\Pi)^{1+\frac{2}{s}}}\,, \tag{4}$$

the value $G(\Pi)$ is the normalized second moment of Π. For more details we refer to [CS10].

Lattice Quantizer: The set of N points are chosen to be the lattice points in a specific region in $\mathbb{R}^s$, for example the unit cube $[0,1)^s$. The lattice points are defined through a generator matrix and the output of the quantization procedure is now an integer vector ξ.

With Equation (4), the quantization performance of different lattices can be compared. In Table 1 we present $G(\Pi)$ for dimensions 1 to 6 for the Cartesian lattice $\mathbb{Z}^s$, the optimal lattice in the respective dimension, and optimized rank-1 lattices found through computer search. The rank-1 lattices are near the optimal quantization error and are presented in the next section.

4. Rank-1 Lattices

First introduced by Korobov [Kor59], rank-1 lattices have been widely studied since then, especially in the field of numerical analysis and quasi-Monte Carlo integration [HHW81, Nie92]. For their use in computer graphics applications see the references given in the related work, especially [Dam09].

4.1. Definition

The points $\mathbf{x_i}$ of an s-dimensional rank-1 lattice in the unit cube $I^s = [0,1)^s$ are given by

$$L_{n,\mathbf{g}} := \left\{ \mathbf{x_i} := \left\{ \frac{i}{n}\mathbf{g} \right\}_1 \middle| i = 0, ..., n-1 \right\}, \tag{5}$$

where $\mathbf{g} \in \mathbb{N}^s$ is a suitable integer generator vector for a fixed number $n \in \mathbb{N}$ of points. $\{\mathbf{x}\}_1$ is the fractional part of $\mathbf{x}$, i.e. the lattice is restricted to the unit cube resulting in a one-periodic pattern.

A suitable generator vector $\mathbf{g} = (g_1, ..., g_s)$ meets the condition:

$$\mathbf{gcd}(g_1, ..., g_s, n) = 1\,, \tag{6}$$

where $\mathbf{gcd}(\cdot)$ is the greatest common divisor of all the generator vector components and the number of points (i.e. the divisor is common for all values, as such the values are not required to be relative prime). If this condition is not met, points would coincide and the first condition of a lattice would be violated (see Section 3.1). When the generator vector has the special form $\mathbf{g} = (1, a, a^2, ...)$ for $a \in [2, n-1]$, the lattice is a *Korobov rank-1 lattice*. This is the only form we use in this paper as it simplifies the search for a good lattice by restricting the search space and allows easy computation of the index i of a lattice point $\mathbf{x_i}$ in s dimensions (see Section 4.4). Other than that, the rank-1 lattice search in Section 4.3 is independent of that choice.

Note that rank-1 lattices have two main advantages over other lattices. First, they exist for any number of points in the respective domain, i.e. the unit cube. Second, encoding a lattice point is simply done through its index i. This gives us the important advantage to quantize to any number of bits desired. Examples for two-dimensional rank-1 lattices can be seen in Figure 1.

4.2. Choosing good Rank-1 Lattices

Not every choice of generator vector leads to a rank-1 lattice that is suitable for quantization, as can be seen in the first image of Figure 1. We therefore propose *maximized minimum distance* (MMD) rank-1 lattices for quantization. This follows the same argumentation as for the two dimensional case in [DDKL09]. For higher dimensions maximization of the minimal distance leads to minimizing the maximal di-

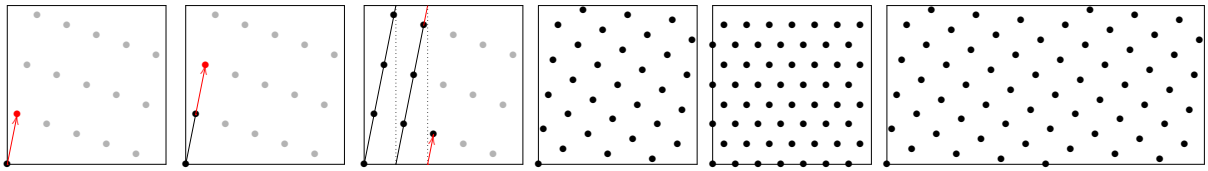


Figure 1: *The first three drawings show the rank-1 lattice $L_{16,(1,5)}$ and how it is constituted by the generator vector. The last three show the maximized minimum distance rank-1 lattices $L_{32,(1,7)}$, $L_{56,(4,7)}$, and $L_{64,(1,14)}$. The rightmost lattice, however, is MMD-optimized for the region $[0,2)\times[0,1)$ instead of $[0,1)^2$, see Section 4.5.*

ameter of the Voronoi cells of the lattice and as such directly reduces the quantization error in Equation 4.

To illustrate this, and verify that there are actually rank-1 lattices in higher dimensions with small quantization error, we plotted quantization error graphs in Figure 2 for random generator vectors. The minimum distance is plotted versus the quantization error $G(\Pi)$ of the corresponding lattice. It can be seen that a large minimum distance results in a smaller error, even in higher dimensions.

Since there is no known method to directly construct generator vectors of MMD rank-1 lattices, the generator vectors need to be determined by a computer search. Methods for two dimensions can be found in [DDK08]. In the following we show how these methods can be generalized to higher dimensions.

4.3. MMD Rank-1 Lattice Search

The basic procedure for finding rank-1 lattices with (nearly optimal) maximized minimum distance consists of the following steps:

1. Choose candidate generator vector.
2. Construct a lattice basis out of this generator vector.
3. Apply a basis reduction algorithm.
   $\rightarrow$ This leads to the shortest vector in the lattice.
4. Termination: goto step 1 until "good" lattice found.

Basically, these steps are the same as for the two dimensional case. Only the steps two and three need to be modified. A python implementation of the computer search can be found in the supplemental material.

Choose Candidate Generator Vector: To restrict the search space, we use generator vectors in Korobov form $\mathbf{g}=(1,a,a^2,...)$ for $a\in[2,n-1]$. The a value is enumerated, or chosen at random. The length of the generator vector is a trivial upper bound for the actual minimum distance of the lattice, as such, short candidates can be skipped before actual basis reduction. Note, that the following algorithms also work for general generator vectors as defined in Section 4.1.

Construct Initial Lattice Basis: Constructing a lattice basis out of a given generator vector for two dimensions is shown by [Rot97]. Apparently an s-dimensional rank-1 lattice $L_{n,\mathbf{g}}$ can be constructed by $s+1$ vectors $\mathbf{u_1},...,\mathbf{u_{s+1}}$, where $\mathbf{u_1}:=\mathbf{g}$ and $\mathbf{u_{i+1}}=n\cdot\mathbf{e_i}$ ($i=1,...,s$ and $\mathbf{e_i}$ denotes the i-th unit vector in s-dimensions). The idea now is to construct an unimodular matrix D_{s+1} (square integer matrix with determinant $+1$ or -1), such that

$$(\mathbf{u_1},...,\mathbf{u_{s+1}})D_{s+1}=(\mathbf{0},\mathbf{b_1},...,\mathbf{b_s})\,, \tag{7}$$

where $\mathbf{b_1},...,\mathbf{b_s}$ is the demanded initial basis.

Let $\alpha=(\alpha_1,...,\alpha_{s+1})$ be the first column of that matrix. Then

$$\alpha_1\mathbf{u_1}+...+\alpha_{s+1}\mathbf{u_{s+1}}=\mathbf{0}\,, \tag{8}$$

is demanded by Equation (7) and as such α can be set to $\alpha_1=n,\alpha_2=-g_1,...,\alpha_{s+1}=-g_s$. Note that $\mathbf{gcd}(\alpha_1,...,\alpha_{s+1})=\mathbf{gcd}(\mathbf{g},n)=1$ (see Equation (6)). Given only the single integer column vector α of the matrix D_{s+1}, the matrix can be completed to the full unimodular matrix [New72]. D_2 is given by:

$$D_2=\begin{pmatrix}\alpha_1 & \sigma\\ \alpha_2 & \rho\end{pmatrix}\,, \tag{9}$$

where the integer elements ρ and σ can be determined via the extended Euclidean algorithm [Knu73]:

$$\rho\alpha_1-\sigma\alpha_2=\delta_2\,. \tag{10}$$

Note that $\delta_2=\mathbf{gcd}(\alpha_1,\alpha_2)$ and is equal to the determinant of the matrix D_2. By the following scheme, D_s can be completed recursively to the full unimodular matrix D_{s+1}:

$$D_{s+1}=\begin{pmatrix} & D_s & & \frac{\alpha_1\sigma}{\delta_s}\\ & & & \frac{\alpha_2\sigma}{\delta_s}\\ & & & \vdots\\ & & & \frac{\alpha_s\sigma}{\delta_s}\\ \alpha_{s+1} & 0 & \cdots & 0 & \rho\end{pmatrix}\,, \tag{11}$$

where ρ and σ are now determined such that $\rho\delta_s-\sigma\alpha_{s+1}=\delta_{s+1}$, with $\delta_{s+1}=\mathbf{gcd}(\alpha_1,...,\alpha_{s+1})$. Note that the last column results in an integer vector despite the division operation, as δ_s is an integer factor of $\alpha_1,...,\alpha_s$.

Basis Reduction: For the basis reduction step during the rank-1 lattice search we use the algorithm given in [NS04]

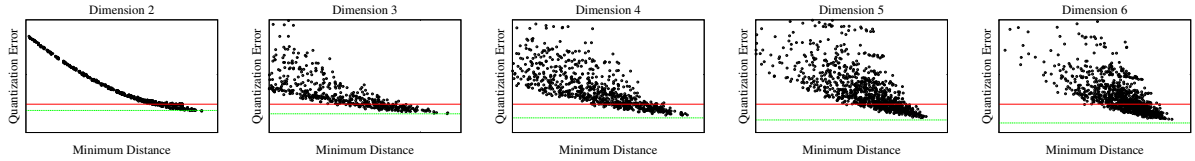


Figure 2: *Correlation between minimum distance and quantization error plotted for dimensions* 2 *to* 6*. Each dot in the plots represents the minimum distance and quantization error of an s-dimensional rank-1 lattice with* 1024 *points, where the generator vector is chosen at random. In all plots the y-axis (quantization error) covers the range* $[0.07, 0.14]$*. Note that both axes are at logarithmic scale. The red line shows the quantization error for the Cartesian lattice* $\mathbb{Z}^s$ *(always* 0.08333*) and the green line the quantization error for the best known lattice quantizer for the corresponding dimension (compare this also to Table 1). The quantization error of the rank-1 lattice was computed via Monte-Carlo integration. It can be seen that maximizing the minimum distance of a rank-1 lattice leads to a small quantization error.*

as this algorithm works fully in integer arithmetic (i.e. no round-off errors) and finding the reduced basis for dimensions up to six is guaranteed. The algorithm computes the *Minkowski-reduced* basis, which has the important property that in the reduced basis $\mathbf{b_1}$ corresponds to the shortest vector in the lattice and determines the minimum distance between the lattice points.

Algorithm 1 Greedy basis reduction

Require: Ordered basis $[\mathbf{b_1}, ..., \mathbf{b_s}]_{\leq}$, $\| \mathbf{b_1} \| \leq ... \leq \| \mathbf{b_s} \|$
Ensure: Minkowski-reduced basis $[\mathbf{b_1}, ..., \mathbf{b_s}]_{\leq}$
  $k \leftarrow 2$
  while $k \leq s$ **do**
    Compute a vector $\mathbf{c} \in L[\mathbf{b_1}, ..., \mathbf{b_{k-1}}]$ closest to $\mathbf{b_k}$
    $\mathbf{b_k} \leftarrow \mathbf{b_k} - \mathbf{c}$
    if $\| \mathbf{b_k} \| \geq \| \mathbf{b_{k-1}} \|$ **then**
      $k \leftarrow k + 1$
    else
      inset $\mathbf{b_k}$ at his length rank k' (vectors are sorted $\leq$)
    end if
    $k \leftarrow k' + 1$
  end while

Computing the vector $\mathbf{c}$ is done by solving a linear equation system,

$$\mathbf{b_k} = \xi[\mathbf{b_1}, ..., \mathbf{b_{k-1}}] \,, \tag{12}$$

and clamping the resulting coefficients ξ_i of the vector ξ to integer values. Then $\mathbf{c}$ is chosen according to:

$$\mathbf{c} = (\xi + \Delta)[\mathbf{b_1}, ..., \mathbf{b_{k-1}}] \,, \text{ minimize } \| \mathbf{c} - \mathbf{b_k} \| \,, \tag{13}$$

where Δ is an s-dimensional vector, where each component Δ_i is either 0 or 1 and all possible 2^s combinations are enumerated for finding the minimum of (13). For dimension $s \geq 5$, the closest vector computation needs to be altered in order to find the reduced basis, instead compute $\mathbf{c} \in L[\mathbf{b_1}, ..., \mathbf{b_{k-1}}, \mathbf{b_{k+1}}, ..., \mathbf{b_s}]$ closest to $\mathbf{b_k}$. For further details and proof of correctness, we refer to [NS04].

Termination: As quality of a candidate generator vector, we use exclusively the minimum distance of the lattice. In all our test, we found a lattice with large minimum distance in less than 2^{16} iterations. These lattices had always good quantization properties, near the optimal.

For early termination, the following two strategies can be used. The first strategy is to compute the ratio of the minimum distance l to the minimum distance l_{min} of a (hypothetical) Cartesian lattice with n points in the unit cube:

$$l_{min} = \sqrt[s]{\frac{1}{n}}. \tag{14}$$

Alternatively, the ratio to an upper bound can be used. As noted in [Dam09], the minimum distance in a densest sphere packing lattice can be used as an upper bound for the maximal minimum distance of rank-1 lattices. In contrast to [Dam09], we use the center density of a lattice [CS10] to derive this upper bound:

$$\delta = \rho^n \cdot (det\ \mathbf{B})^{-\frac{1}{2}}, \tag{15}$$

where ρ is the radius of one sphere. As $(det\ \mathbf{B})^{\frac{1}{2}} = Vol(\Pi) = \frac{1}{n}$ for a rank-1 lattice with n points in the unit cube, and with $l = 2\rho$, we get an upper bound for the minimum distance:

$$l = 2 \cdot \sqrt[s]{\frac{\delta}{n}} \leq l_{max} = 2 \cdot \sqrt[s]{\frac{\delta_{max}}{n}}, \tag{16}$$

where δ_{max} is the center density of the densest lattice sphere packing in the respective dimension:

s	2	3	4	5	6
δ_{max}	$\frac{1}{2\cdot\sqrt{3}}$	$\frac{1}{4\cdot\sqrt{2}}$	$\frac{1}{8}$	$\frac{1}{8\cdot\sqrt{2}}$	$\frac{1}{8\cdot\sqrt{3}}$

As generic limits, for lattices with $n \geq 2^{16}$, we suggest the following limits:

s	2	3	4	5	6
l/l_{min}	1.07	1.08	1.09	1.12	1.09
l/l_{max}	0.99	0.96	0.92	0.91	0.84

	Single Core		12 Cores	
s	Quant.	Recon.	Quant.	Recon.
2	4.22	0.574	1.61	0.089
3	5.71	0.538	1.62	0.089
4	13.45	0.536	1.84	0.086
5	24.23	0.527	2.22	0.086
6	45.32	0.518	4.02	0.084

Table 2: *Time measurements (in seconds) for quantization and reconstruction of 1 GB of double precision data with a straight forward C++ implementation of the presented algorithms. Measurements where taken on a computer with two Intel Xeon X5660 6 Core CPUs with 2.80 GHz. The cost of the reconstruction step is clearly independent of the dimension (one multiplication and one modulo operation per component (5)). The quantization step, however, shows an exponential increase with higher dimensions, but still in a practical range. Other than that, it scales perfectly with multiple cores as each s-dimensional vector can be quantized independently. Quantization bit depth was set to* 48 *bits.*

4.4. Quantization with a Rank-1 Lattice

Quantizing a vector $\mathbf{v} \in [0,1)^s$ is done by first solving the linear equation system

$$\mathbf{v} = \xi[\mathbf{b_1}, ..., \mathbf{b_s}] \,, \tag{17}$$

and clamping the resulting coefficients of ξ to integer values (compare to (12)). This results in the Cartesian coordinates of a lattice point $\mathbf{p}$. The lattice point $\mathbf{p}$ is the anchor point of a parallelepiped given by,

$$(\xi + \Delta)[\mathbf{b_1}, ..., \mathbf{b_s}] \,, \text{ where } \Delta \in [0,1)^s \,, \tag{18}$$

containing no other lattice point but the point $\mathbf{v}$ (compare to (13)). By enumerating all corner points of the parallelepiped the nearest lattice point $\mathbf{p}_{near}$ is found. This is in fact very similar to the previews method for finding the vector $\mathbf{c}$ during basis reduction.

What remains is to translate the Cartesian coordinates of $\mathbf{p}_{near}$ into its index i_{near} in the rank-1 lattice (5). If the first component of the generator vector g_1 is equal to 1 (which is the case for $\mathbf{g}$ in Korobov form), then the index i of the lattice point $\mathbf{p}$ is given by its first component $i = p_1$ (this directly follows from (1) and (5)). Hence, for a rank-1 lattice with 2^b points, we need b-bits to store i, which directly translates to the quantized vector via the given generator vector $\mathbf{g}$.

Note that computing the closest lattice point (also referred to as Closest Vector Problem, CVP) is in general NP-hard for high dimensions, even if the reduced basis is known [AEVZ00, HS07]. However, for dimensions up to 6 this is not severe as can be seen in Table 2 where we measured the performance of this procedure. Also see the discussion in Section 6.3.

4.5. Quantization on the Hyperrectangle

As data is often not equally wide distributed in all dimensions, it is desirable to have the quantization method not be restricted to a hypercube. Quantizing data on a hyperrectangular region $[0,x_1) \times [0,x_2) \times ... \times [0,x_s)$ with a given rank-1 lattice is strait forward. The points of the rank-1 lattice inside the hypercube $[0,1)^s$ need to be scaled to match the hyperrectangle, this is done by simply scaling the lattices basis before solving for Equation (17). To compute the index of the lattice point, the point needs to be scaled back to the $[0,1)^s$ domain. The other steps of quantization remain the same as in Section 4.4.

What remains is to optimize the rank-1 lattice for maximized minimum distance on the scaled domain (see Figure 1, rightmost lattice). This is achieved by scaling the initial basis (7) before basis reduction, as shown by Dammertz [Dam09]. This results in a rank-1 lattice which is optimized for the scaled domain. See Figure 5 where this method is applied to the Stanford Buddha.

5. Practical Application of Rank-1 Quantizers

In this section we present a step by step guide on how to use our proposed quantization scheme in practical applications. We assume that the input data is in the unit cube $[0,1)^s$, or on a hyperrectangle. A python implementation of the quantization steps can be found in the supplemental material.

For ease of reference here is a summary of the used symbols

s	dimension of the data
b	quantization bit-depth (i.e. $32 bits$)
n	$n = 2^b$, number of rank-1 lattice points
$\mathbf{g}$	the generator vector
i	index of a data-point (represented by b bits)
d_i	the corresponding datapoint to index i

For the preparation and quantization of the input data three steps need to be performed:

A) Bit-Depth and Dimension First the dimensionality of the input data needs to be known and the target bit-depth needs to be defined.

B) Choose Generator Vector Next, the corresponding optimal generator vector for the rank-1 lattice needs to be selected. This can be done by either looking it up in a table (a shortened version of this data can be found in Table 3) or by using the provided python implementation of the search from Section 4.3.

C) Quantization Finally all data points need to be quantized by projection onto the corresponding basis and the resulting integer point index i is the quantized representation of this data-point, see Section 4.4 and 4.5.

To use the quantized data only a simple computation needs to be performed:

D) Reconstruct Reconstructing the quantized data from the

index is now done by a simple multiplication with the generator vector and a modulo operation:

$$d_i = \frac{1}{n}(i\,\mathbf{g} \mod n)\,.$$

This can be efficiently implemented on any current hardware architecture when n is chosen as a power of two with a bit-wise `and` operation instead of the modulo. Note that the result needs to be additionally scaled when the quantization was performed on a domain other than the unit cube $[0,1)^s$.

$s\backslash b$	16	32	48	64
2	25962	306994	221529218	4615321823
3	23128	264892	2697033	365423322
4	20166	234139	373961	16832394
5	2616	191538	143618	1494331
6	20836	57697	12284	227042

Table 3: *List of the (a) parameter of the Korobov generator vectors $\mathbf{g} = (1, a, a^2, \ldots)$ for near optimal rank-1 lattices for selected bit-depths b and dimensions s.*

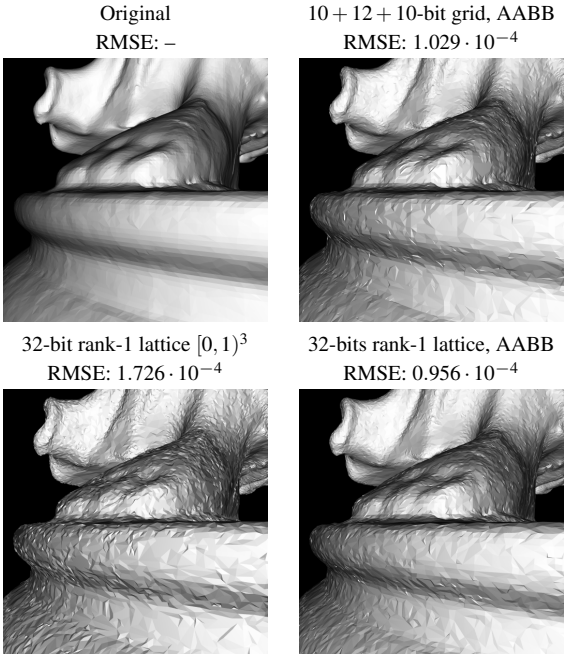


Figure 5: *Stanford Buddha quantized to 32-bits per vertex. The configuration of $10 + 12 + 10$-bits is optimal for regular grid quantization of this model. However, rank-1 lattice quantization shows better results than the regular grid when optimized for the axis aligned bounding box. Finding the parameters for this lattice took about 2 minutes with the python implementation of the search algorithm.*

6. Example Applications and Discussion

Here we show two example applications of ad-hoc quantization of multi-dimensional input data and compare the results to the regular grid quantization. The first application is the quantization of $3d$ vertex position in a triangle mesh. In the second application we quantize $6d$ spectral reflectance textures.

6.1. Mesh Vertex Quantization

A very simple and straightforward application is the quantization of $3d$ positions. Figure 3 shows a closeup of different quantization options used on a mesh and Table 4 shows the quantization error for different input meshes. The quantization error for all three meshes is almost identical even though the meshes themselves are quite different because they have no specific structure and enough vertices to behave almost as random data input regarding the lattices. Quantization using the axis aligned bounding box as domain instead of the unit cube is shown in Figure 5.

6.2. Spectral Reflectance Quantization

Multi-spectral reflectance textures are a good example for the use of rank-1 lattices for quantization because they are naturally high dimensional and require a lot of storage space for high resolutions. Figure 4 shows an example of quantizing a $6d$ spectral texture and the corresponding errors. The spectral reflectance textures we use here were captured in a project [RSK10] at the University of Bonn. It can be seen that the rank-1 lattice performs as well as the theoretical considerations in Section 4 predicted. For more comparison images and bit-depths see the supplemental material.

6.3. Discussion

The quantization error reported in the examples above is a direct result of the theoretical derivation and construction from Section 4. This shows that similar results can be expected independent of the application domain as long as the input data is unstructured (i.e. not already on a regular grid).

One important possible drawback of using rank-1 lattices for quantization is inherent in the irregular structure. When the input data contains the same value across all dimensions a rank-1 lattice will produce slightly different values for each dimension while the regular grid (with equal number of bits per dimension) will produce the same value for all dimensions. For example the vector $(0.3, 0.3, 0.3)$ is quantized to $(0.25, 0.25, 0.25)$ using a regular grid with 3×2-*bit* and $(0.26, 0.28, 0.25)$ with a 6-*bit* rank-1 lattice.

Another aspect of rank-1 lattices in the form we presented above are the lack of any simple hierarchical structure on them. It is thus not easily possible to embed finer lattices into parts of the domain as can be done with regular grids and was done for example by Segovia et al. [SE10] and Garanzha et al. [GBG11] for ray tracing meshes. The use of

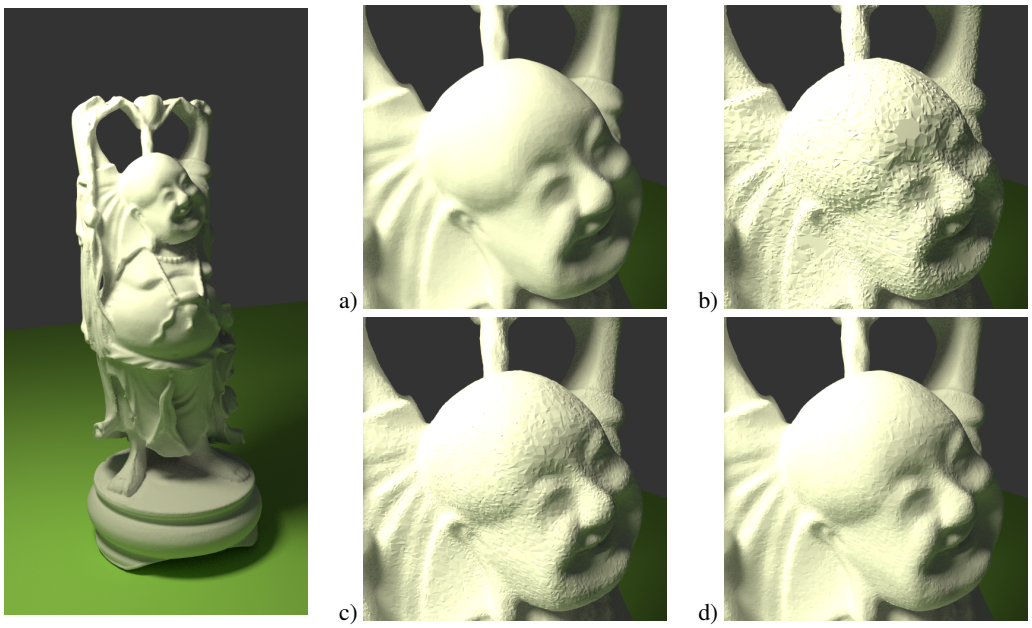


Figure 3: *This figure shows a direct comparison of different quantization bit depths for rank-1 lattices. a) shows the original Stanford Buddha b) quantized to* 30*-bit c)* 33*-bit d)* 36*-bit. The bit-depths here were chosen to be directly comparable to a regular grid quantization. For more and different bit-depths see Table 4.*

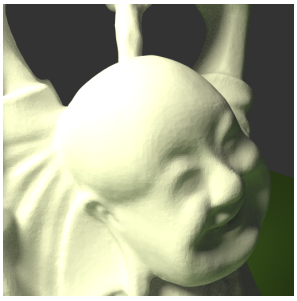
	Buddha 	**Museum** 	**Conference** 
# Triangles	1087542	1468284	1064498
regular grid (30-*bit*)	$2.82 \cdot 10^{-4}$	$2.84 \cdot 10^{-4}$	$2.77 \cdot 10^{-4}$
regular grid (33-*bit*)	$1.41 \cdot 10^{-4}$	$1.41 \cdot 10^{-4}$	$1.41 \cdot 10^{-4}$
regular grid (36-*bit*)	$7.04 \cdot 10^{-5}$	$7.00 \cdot 10^{-5}$	$7.04 \cdot 10^{-5}$
r1-Lattice (30-*bit*)	$2.74 \cdot 10^{-4}$	$2.74 \cdot 10^{-4}$	$2.74 \cdot 10^{-4}$
r1-Lattice (32-*bit*)	$1.73 \cdot 10^{-4}$	$1.73 \cdot 10^{-4}$	$1.72 \cdot 10^{-4}$
r1-Lattice (33-*bit*)	$1.37 \cdot 10^{-4}$	$1.37 \cdot 10^{-4}$	$1.37 \cdot 10^{-4}$
r1-Lattice (36-*bit*)	$6.85 \cdot 10^{-5}$	$6.86 \cdot 10^{-5}$	$6.86 \cdot 10^{-5}$
r1-Lattice (37-*bit*)	$5.44 \cdot 10^{-5}$	$5.44 \cdot 10^{-5}$	$5.44 \cdot 10^{-5}$

Table 4: *This table shows the RMSE quantization error of different lattice configurations applied to three differen input meshes.*

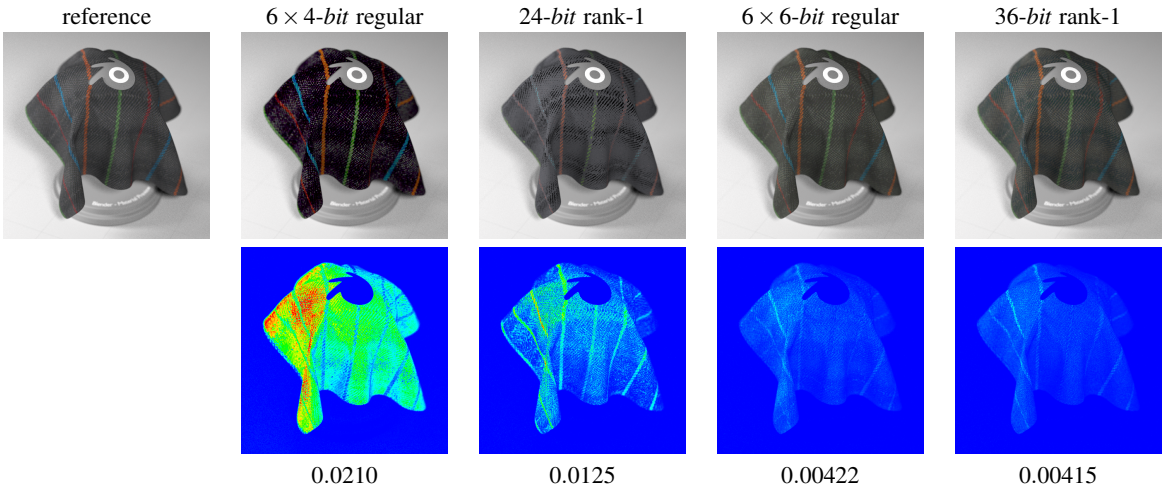


Figure 4: *This figure shows the result of quantizing a* 6*d spectral reflectance texture to* 24*-bit and to* 36*-bit using a regular grid and a rank-1 lattice. The second row shows the difference to the reference image and the last row the RMSE of the quantized texture data. The input data has a resolution of* $832 \times 669 \times 6$ *(*12.7 *MB) and the quantization results in* 1.6 *MB and* 2.1 *MB respectively.*

rank-1 lattice sequences [HHLL00] could solve this problem but significantly increase the complexity of implementation and search.

Our proposed method is limited by the exponential complexity of basis reduction and quantization with respect to the dimension (Section 4.4). However, for dimensions up to six we have shown the practicability of this approach. Furthermore, the reconstruction is always simple and independent of the dimension, which is a main advantage over other lattices. A general lattice is defined by a generator matrix (i.e. the lattice basis) and not just a single generator vector, which makes reconstruction more difficult and further needs special care when encoding the coefficients.

7. Future Work

As can be seen in Table 1 and Figure 2, the gab to the optimal quantization lattice gets larger with higher dimensions, though, the advantage to the $\mathbb{Z}^s$ lattice is still improving. This gab needs to be further investigated, for example, does it substantially improve when not restricted to Korobov rank-1 lattices?

Another topic is to improve the shortest vector search and the basis reduction. They may be improved trough generic algorithm, such as [LLM06], or through exploiting the special arithmetic structure of rank-1 lattices.

Furthermore, other applications for these lattices can be investigated, such as sampling and discretization of multidimensional domains.

8. Conclusion

We have demonstrated the theoretical concept of using rank-1 lattices as simple quantizers for practical applications. The main advantages of rank-1 lattices are:

- Arbitrary bit-depth b, independent of dimension.
- Close to optimal lattice quantization error (always better than regular grid).
- Very easy and fast to compute the data from the quantized bit representation (s multiplications and bit masking).
- Same code and structure for any dimension s.

Our experiments have shown that the construction principle for rank-1 lattices we derived result in lattices that always outperform the commonly used regular grid while being still almost as easy to use. We also showed that these resulting rank-1 lattices are close to the optimal lattice for their respective dimension. The advantage of the ability to quantize to arbitrary bit-depth allows for example to quantize 3d data directly into 32-$bits$ which fits current computer architectures well. This flexibility is achieved without any special cases by just choosing a different generator vector and the target bit resolution can be fine-tuned for the current application.

Acknowledgments

We thank the reviewers for their helpful comments. This work has been partially funded by the DFG Emmy Noether fellowship (Le 1341/1-1) and by an NVIDIA Professor Partnership Award. The Happy Buddha model is from the Stanford Scanning Repository. The museum scene was modeled by Alvaro Luna Bautista and Joel Andersdon. The cloth scene was modeled by Robin Marín. The conference room was modeled by Anat Grynberg and Greg Ward.

References

[ABD10] ADAMS A., BAEK J., DAVIS M. A.: Fast High-Dimensional Filtering Using the Permutohedral Lattice. *Comput. Graph. Forum 29*, 2 (2010), 753–762. 272

[AEVZ00] AGRELL E., ERIKSSON T., VARDY A., ZEGER K.: Closest Point Search in Lattices. *IEEE TRANS. INFORM. THEORY 48* (2000), 2201–2214. 276

[CS10] CONWAY J., SLOANE N.: *Sphere Packings, Lattices and Groups*. Grundlehren Der Mathematischen Wissenschaften. Springer, 2010. 272, 273, 275

[Cse05] CSEBFALVI B.: Prefiltered Gaussian Reconstruction for High-Quality Rendering of Volumetric Data sampled on a Body-Centered Cubic Grid. In *IEEE Visualization* (2005), IEEE Computer Society, p. 40. 272

[CVFH08] CONDAT L., VILLE D. V. D., FORSTER-HEINLEIN B.: Reversible, Fast, and High-Quality Grid Conversions. *IEEE Transactions on Image Processing 17*, 5 (2008), 679–693. 272

[Dam09] DAMMERTZ S.: *Rank-1 Lattices in Computer Graphics*. PhD thesis, Ulm University, 2009. 272, 273, 275, 276

[DDK08] DAMMERTZ S., DAMMERTZ H., KELLER A.: Efficient Search for Low-Dimensional Rank-1 Lattices with Applications in Graphics. In *Proc. Monte Carlo and Quasi-Monte Carlo Methods 2006*. Springer, 2008, pp. 217–236. 272, 274

[DDKL09] DAMMERTZ S., DAMMERTZ H., KELLER A., LENSCH H. P. A.: Textures on Rank-1 Lattices. *Comput. Graph. Forum 28*, 7 (2009), 1945–1954. 272, 273

[DK08] DAMMERTZ S., KELLER A.: Image Synthesis by Rank-1 Lattices. Monte Carlo and quasi-Monte Carlo methods 2006, selected papers, 2008. 272

[EVDVM08] ENTEZARI A., VAN DE VILLE D., MOLLER T.: Practical Box Splines for Reconstruction on the Body Centered Cubic Lattice. *Visualization and Computer Graphics, IEEE Transactions on 14*, 2 (march-april 2008), 313 –328. 272

[GBG11] GARANZHA K., BELY A., GALAKTIONOV V.: Simple Geometry Compression for Ray Tracing on GPU. In *Conference proceedings of 21-th International Conference on Computer Graphics and Vision GraphiCon-2011* (2011), pp. 107 – 110. 277

[GG91] GERSHO A., GRAY R. M.: *Vector quantization and signal compression*. Kluwer Academic Publishers, Norwell, MA, USA, 1991. 272

[HHLL00] HICKERNELL F. J., HONG H. S., LÉCUYER P., LEMIEUX C.: Extensible Lattice Sequences for Quasi-Monte Carlo Quadrat. *SIAM J.Sci.Comput. 22*, 3 (Mar. 2000), 1117–1138. 279

[HHW81] HUA L., HUA L., WANG Y.: *Applications of Number Theory to Numerical Analysis*. Springer-Verlag, 1981. 273

[HS07] HANROT G., STEHLÉ D.: *Improved Analysis of Kannan's Shortest Lattice Vector Algorithm*. Tech. rep., In Proceedings of Crypto 2007, 2007. 276

[Knu73] KNUTH D. E.: *The Art of Computer Programming, Volume I: Fundamental Algorithms, 2nd Edition*. Addison-Wesley, 1973. 274

[Kor59] KOROBOV N. M.: The Approximate Computation of Multiple Integrals. *Dokl. Akad. Nauk SSSR 124* (1959), 1207–1210. in Russian. 273

[LLM06] LIU Y.-K., LYUBASHEVSKY V., MICCIANCIO D.: On bounded distance decoding for general lattices. In *APPROX-RANDOM* (2006), vol. 4110 of *Lecture Notes in Computer Science*, Springer, pp. 450–461. 279

[LLWQ13] LI B., LI X., WANG K., QIN H.: Surface mesh to volumetric spline conversion with generalized polycubes. *IEEE Transactions on Visualization and Computer Graphics 19*, 9 (2013), 1539–1551. 272

[LQ11] LI B., QIN H.: Feature-Aware Reconstruction of Volume Data via Trivariate Splines. In *Pacific Conference on Computer Graphics and Applications - Short Papers* (2011), Chen B.-Y., Kautz J., Lee T.-Y., Lin M. C., (Eds.), Eurographics Association, pp. 49–54. 272

[MS05] MIDDLETON L., SIVASWAMY J.: *Hexagonal Image Processing: A Practical Approach*. Advances in Pattern Recognition. Springer-Verlag New York, 2005. 272

[New72] NEWMAN M.: *Integral Matrices*. Pure and Applied Mathematics. Academic Press, 1972. 274

[Nie92] NIEDERREITER H.: *Random Number Generation and Quasi-Monte Carlo Methods*. Cbms-Nsf Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1992. 273

[NM02] NEOPHYTOU N., MUELLER K.: Space-Time Points: 4d Splatting on Efficient Grids. In *Proceedings of the 2002 IEEE symposium on Volume visualization and graphics* (Piscataway, NJ, USA, 2002), VVS '02, IEEE Press, pp. 97–106. 272

[NS04] NGUYEN P. Q., STEHLÉ D.: Low-Dimensional Lattice Basis Reduction Revisited. In *Proceedings of the 6th International Algorithmic Number Theory Symposium, (ANTS-VI)* (2004), vol. 3076 of *LNCS*, Springer, pp. 338–357. 274, 275

[Rot97] ROTE G.: Finding a Shortest Vector in a Two-Dimensional Lattice Modulo M. *Theoretical Computer Science 172* (1997), 303–308. 274

[RSK10] RUMP M., SARLETTE R., KLEIN R.: Groundtruth Data for Multispectral Bidirectional Texture Functions. In *CGIV 2010* (June 2010), Society for Imaging Science and Technology, pp. 326–330. 277

[SE10] SEGOVIA B., ERNST M.: Memory Efficient Ray Tracing with Hierarchical Mesh Quantization. In *Graphics Interface 2010* (2010), pp. 153–160. 277

[VVPL02] VAN DE VILLE D., VAN DE WALLE R., PHILIPS W., LEMAHIEU I.: Image Resampling between Orthogonal and Hexagonal Lattices. In *Image Processing. 2002. Proceedings. 2002 International Conference on* (2002), vol. 3, pp. III–389 – III–392 vol.3. 272