Optimal Regular Volume Sampling

Thomas Theußl* Torsten Möller† Meister Eduard Gröller*

*Institute of Computer Graphics and Algorithms
Vienna University of Technology

†Graphics, Usability, and Visualization Lab
Simon Fraser University

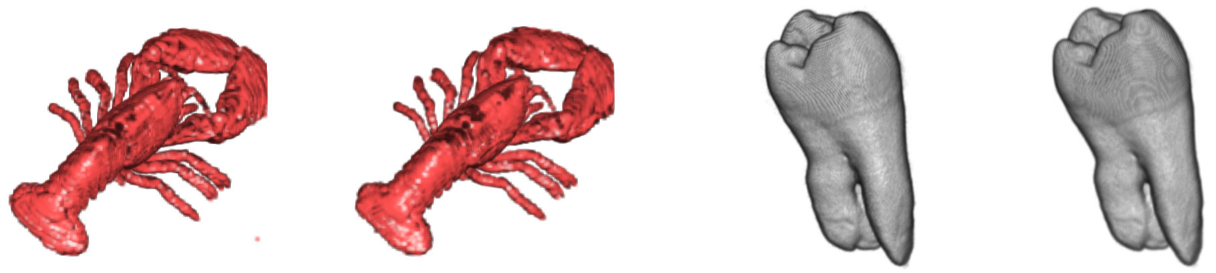


Figure 1: CT scans of a lobster and a tooth, represented on Cartesian and body-centered cubic grids (left and right images, respectively). The representation via body-centered cubic grids requires approximately 30% less samples.

Abstract

The classification of volumetric data sets as well as their rendering algorithms are typically based on the representation of the underlying grid. Grid structures based on a Cartesian lattice are the de-facto standard for regular representations of volumetric data. In this paper we introduce a more general concept of regular grids for the representation of volumetric data. We demonstrate that a specific type of regular lattice – the so-called *body-centered cubic* – is able to represent the same data set as a Cartesian grid to the same accuracy but with 29.3% fewer samples. This speeds up traditional volume rendering algorithms by the same ratio, which we demonstrate by adopting a splatting implementation for these new lattices. We investigate different filtering methods required for computing the normals on this lattice. The lattice representation results also in lossless compression ratios that are better than previously reported. Although other regular grid structures achieve the same sample efficiency, the body-centered cubic is particularly easy to use. The only assumption necessary is that the underlying volume is isotropic and band-limited - an assumption that is valid for most practical data sets.

Keywords: volume data, Cartesian grid, close packing, hexagonal sampling, body centered cubic

*{theussl,meister}@cg.tuwien.ac.at, http://www.cg.tuwien.ac.at/home/
†torsten@cs.sfu.ca, http://www.cs.sfu.ca/~graphics/

1 Introduction

Different grid structures have been studied extensively in various fields like chemistry [18], solid state physics [1], condensed matter physics [10], or crystallography [7]. Researchers in these fields have studied the structure of atoms and molecules, which are often placed in a regular grid structure. This structure is optimized for energy states [1] and results in covering space as closely as possible.

Results in multi-dimensional signal processing show that a Cartesian sampling structure is not the most efficient one [16]. Efficiency here is measured in terms of sampling points per unit hyper-volume. Under the assumption that the sampled function is isotropic and band-limited the resulting frequency support would be a hyper-sphere. Hence the most efficient sampling scheme would arrange the replicated (hyper-spherical) frequency response as densely as possible in frequency domain.

The problem of how to place as many (hyper-)spheres as possible in a fixed (hyper-)volume is known as the sphere packing problem [14]. This has been studied by many mathematicians in up to quite staggering dimensions (Conway and Sloane [3] give examples of dimensions up to 1048584). The problem of packing spheres optimally was stated in 1900 by Hilbert as his now famous Problem 18 [6]. It is still not solved completely. However, several *regular* grid structures are known which are optimal. Among these is the body-centered cubic (bcc) grid, which turns out to be particularly easy to use.

In the image processing community it is well known that sampling an image on a Cartesian grid is not optimal. By using a hexagonal sampling scheme one can save 13.4% of the samples [12]. Research has been directed to adapt algorithms like straight line generation [9], distance transformations [2], or oversampling [8] to hexagonal grids. However, the Cartesian structure of display devices limits the use of hexagonal grids for image processing so that 2D hexagonal grids are rarely used.

In volume visualization, or generally when dealing with 3D

[1]This is, of course, a serious oversimplification. However, for almost all the elements the lowest energy state is crystalline [10].

Presented at IEEE Visualization 2001
October 21 - October 26, 2001
San Diego Paradise Point Resort, San Diego

functions we are not bound to Cartesian grids. The representation of the function we want to visualize can be chosen arbitrarily since typically only two-dimensional projections of the data set are examined. Since a bcc grid can represent isotropic, band-limited data as accurately as Cartesian grids using 29.3% fewer samples [4][2], the advantages of using a bcc grid are significant.

In this paper we show how to take advantage of hexagonal sampling in volume rendering. We outline and propose solutions for the inherent issues of re-sampling of rectangular grids as well as interpolation and gradient estimation.

The remainder of this paper is organized as follows. We summarize the results of hexagonal sampling in 2D and derive an optimal sampling scheme in 3D in Section 2. In Section 3 we show how the splatting algorithm can be adopted for bcc grids, including storing of the data and gradient estimation. In Section 4 we examine acquisition techniques for data sampled on bcc grids. In Section 5 we present the results of our experiments. Some ideas for future work are presented in Section 6, and we derive conclusions of our studies in Section 7.

2 Baseband Optimal Sampling

Usually, no a priori knowledge is available about the (continuous) underlying functions we are sampling. Therefore, we assume that these functions are isotropic, i.e., that they do not have a preferred direction. Another common assumption is that they are band-limited. Both assumptions together result in the property that the frequency responses of such functions are hyper-spheres (circles in 2D and spheres in 3D, respectively).

Sampling such spherically band-limited functions results in replicating the primary spectrum in the frequency domain [15]. In order to reconstruct the underlying continuous signal perfectly, we need to ensure that the samples in spatial domain are close enough, so that the aliased spectra in the frequency domain do not overlap. *Optimal sampling* is achieved when the number of samples that fulfill this condition is minimal. In 1D this is also known as the *Nyquist* sampling rate. In order to optimally sample in higher dimensions (i.e., to use as few samples as possible), aliased spectra in the frequency domain have to be packed as closely as possible. This problem is known as the *sphere packing problem* [14], which has been extensively studied and solutions for regular packing structures in 2D and 3D exist.

As a motivation and for the sake of simplicity we will first describe our method to find an optimal sampling pattern in 2D before we delve into the mysterious structure of 3D Euclidian space.

2.1 Optimal sampling density in 2D

We will describe sampling as a mapping from indices to actual sample positions [4]:

$$\begin{pmatrix} x \\ y \end{pmatrix} = V \cdot \begin{pmatrix} i \\ j \end{pmatrix} \tag{1}$$

Here the integers i, j are the indices of the sample and x, y is the corresponding sampling position. The matrix V, which is called *sampling matrix*, describes the mapping itself, e.g.,

$$V_{rect2D} = \begin{pmatrix} T_1 & 0 \\ 0 & T_2 \end{pmatrix} \tag{2}$$

is the matrix for rectangular sampling which simplifies to the commonly used regular (Cartesian) sampling for $T_1 = T_2$. Hexagonal sampling is most conveniently described by the matrix

$$V_{hex2D} = \begin{pmatrix} T_1 & \frac{1}{2}T_1 \\ 0 & T_2 \end{pmatrix} \qquad \text{for } T_2 = \frac{\sqrt{3}}{2}T_1 \tag{3}$$

which virtually just performs a shear of the rectangular samples followed by a for-shortening. When V describes the sampling in spatial domain, the matrix U, satisfying

$$U^\top V = 2\pi I \tag{4}$$

with $U^\top$ being the transpose of U and I being the identity matrix, describes the positions of the replicas in frequency domain. U is therefore called the *periodicity matrix* [4]. Applying Eq. 4 to Eq. 2 we obtain the periodicity matrix for rectangular sampling

$$U_{rect2D} = \begin{pmatrix} u_1 & 0 \\ 0 & u_2 \end{pmatrix} \tag{5}$$

with $u_1 = \frac{2\pi}{T_1}$ and $u_2 = \frac{2\pi}{T_2}$. The periodicity matrix for hexagonal sampling is

$$U_{hex2D} = \begin{pmatrix} u_1 & 0 \\ -\frac{1}{2}u_2 & u_2 \end{pmatrix} \qquad \text{for } u_2 = \frac{2}{\sqrt{3}}u_1 \tag{6}$$

where again $u_1 = \frac{2\pi}{T_1}$. The 2D Fourier Transform F of a circularly band-limited signal has the property

$$\mathrm{F}(\omega_1, \omega_2) = 0 \qquad \text{for } \omega_1^2 + \omega_2^2 \geq W^2 \tag{7}$$

where W is the maximum frequency in the data set. This baseband can be inscribed, for example, in a square with length $l = 2W$ (corresponding to rectangular sampling). In other words, u_1 and u_2 in Eq. 5 have to be equal to $2W$. On the other hand, the baseband can also be inscribed in a hexagon with side length $l = \frac{2}{\sqrt{3}}W$ (corresponding to hexagonal sampling). This means that in Eq. 6 u_2 must be equal to $2W$ (see Fig. 2). Calculating the sampling matrices from these periodicity matrices, we end up with

$$V_{rect2D} = \begin{pmatrix} \frac{\pi}{W} & 0 \\ 0 & \frac{\pi}{W} \end{pmatrix} \tag{8}$$

where

$$|\det V_{rect2D}| = \frac{\pi^2}{W^2} \tag{9}$$

and

$$V_{hex2D} = \begin{pmatrix} \frac{2}{\sqrt{3}}\frac{\pi}{W} & \frac{1}{\sqrt{3}}\frac{\pi}{W} \\ 0 & \frac{\pi}{W} \end{pmatrix} \tag{10}$$

where

$$|\det V_{hex2D}| = \frac{\pi^2}{W^2}\frac{2}{\sqrt{3}}. \tag{11}$$

The sampling density is proportional to $\frac{1}{\det V}$ [4]. By taking the ratio

$$\frac{|\det V_{rect2D}|}{|\det V_{hex2D}|} = \frac{\sqrt{3}}{2} = 0.866 \tag{12}$$

we see that hexagonal sampling requires 13.4% fewer samples than rectangular sampling.

[2]In Table 1.1 on page 47 Dudgeon and Mersereau give a sampling density ratio of 0.705, i.e., 29.5% fewer samples. This is simply due to a rounding error, compare the results of Petersen and Middleton [16].

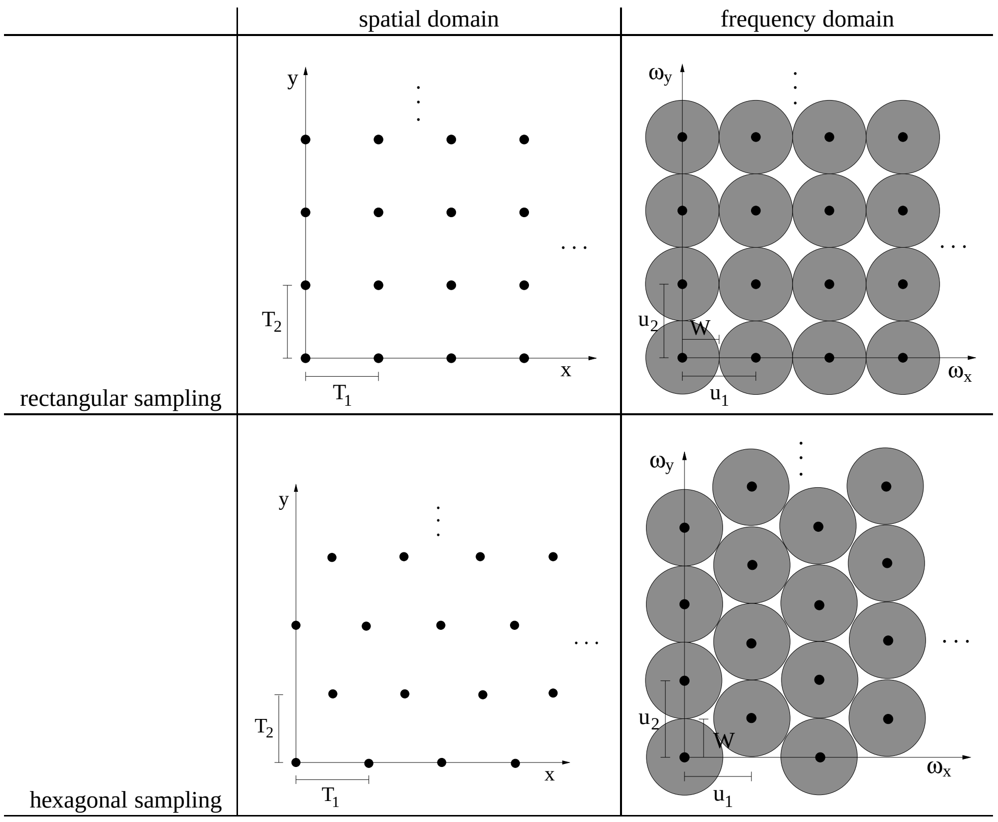


Figure 2: 2D regular rectangular and hexagonal sampling in the spatial and frequency domains.

2.2 Optimal sampling density in 3D

Analogous to the 2D case, we can describe the mapping from indices i, j, k to coordinates x, y, z for rectilinear grids using the following matrix:

$$V_{rect3D} = \begin{pmatrix} T_1 & 0 & 0 \\ 0 & T_2 & 0 \\ 0 & 0 & T_3 \end{pmatrix} \tag{13}$$

which is regular (rectangular) when $T_1 = T_2 = T_3$.

As expected, regular rectangular sampling in 3D is (as in 2D) not optimal. It is important to note that an optimal sphere packing for arbitrary packing structures in 3D is not known. However, several optimal packing structures, all with equal sampling density, are known for the case of regular sampling, i.e., structures that can be described by three base vectors. Fortunately, this is exactly what we need, since we do not want to sacrifice the implicit indexing of the grid points that makes regular grid representations so attractive.

Among the optimal regular packing structures are the hexagonal close packed (hcp) structure and the face centered cubic (fcc) structure [3]. In order to achieve a close packing in the frequency domain using an fcc lattice (the reason why not using an hcp lattice is explained at the end of this section), we use the following matrix:

$$U_{fcc} = U_{fcc}^{\top} = \begin{pmatrix} u & 0 & u \\ 0 & u & u \\ u & u & 0 \end{pmatrix} \tag{14}$$

An fcc lattice consists of simple cubic cells with additional sampling points in the center of each cell face. One cell of an fcc lattice is depicted in Fig. 3 with the base vectors from Eq. 14.

By plugging Eq. 14 in Eq. 4 we end up with a sampling matrix in spatial domain which describes a body centered cubic (bcc) lattice:

$$V_{bcc} = \frac{1}{2} \begin{pmatrix} T & -T & T \\ -T & T & T \\ T & T & -T \end{pmatrix} \tag{15}$$

with $T = \frac{2\pi}{u}$.

A bcc lattice also consists of a simple cubic cell but with only one additional sampling point, which is right in the center of the cell. Fig. 4 depicts one cell of a bcc lattice. Note, that the base vectors of Fig. 4 do not correspond to Eq. 15, as these are rather unintuitive. We chose another set of base vectors, which is more convenient for our purpose of indexing the data points (see Section 3.1). They are described by the sampling matrix

$$V_{bcc} = \begin{pmatrix} T & 0 & \frac{1}{2}T \\ 0 & T & \frac{1}{2}T \\ 0 & 0 & \frac{1}{2}T \end{pmatrix} \tag{16}$$

It is easily verified that the sampling matrices in Eqs. 15 and 16 describe the same set of sample points.

To guarantee that the replicas in frequency domain do not overlap, u must be equal to $2W$ for rectangular sampling. Since the periodicity matrix is analogous to the 2D periodicity matrix we end up with

$$|\det V_{rect3D}| = \frac{\pi^3}{W^3} \tag{17}$$

For the fcc lattice, u must be equal to $\sqrt{2}W$, which yields

$$|\det V_{bcc}| = \frac{\pi^3}{W^3}\sqrt{2} \tag{18}$$

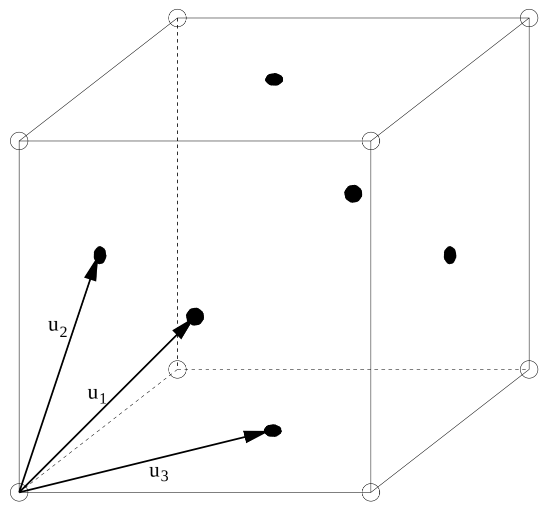


Figure 3: One cell of an fcc lattice with base vectors u_i. The black dots mark additional sample points (in the center of the faces) as compared to a simple cubic cell.

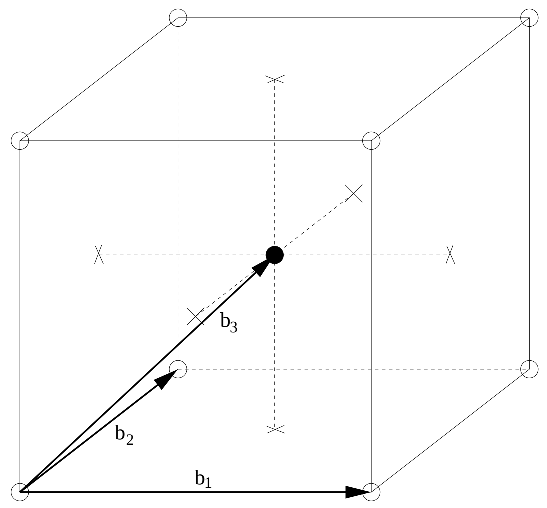


Figure 4: One cell of a bcc structure with base vectors b_i. The only difference to a simple cubic cell is one additional sample point right in the center of the cell, marked with a black dot.

By again taking the ratio

$$\frac{|\det V_{rect3D}|}{|\det V_{bcc}|} = 0.707 \qquad (19)$$

we see that we need 29.3% fewer samples than with rectangular sampling. This means that if we sample a function on a regular rectangular grid with sampling distance T_r, we can increase the sampling distance T_b for a bcc grid to $\sqrt{2}T_r$.

In the above example we started with an fcc lattice in the frequency domain which resulted in a bcc lattice in the spatial domain. We could also choose an hcp lattice, since it has the same packing density as an fcc lattice [3]. However, an hcp lattice in the frequency domain is also an hcp lattice in the spatial domain and hcp lattices are rather difficult to index [7]. Therefore, we prefer to use a bcc grid.

3 Implementation Details

In order to use a bcc grid in practice we have to address some inherent implementation issues. First, we must think about how to organize the grid in memory. We present a scheme which stores the sampling points in a three-dimensional array. The addressing scheme is of special importance, since we want to take advantage of the implicit ordering in regular grid structures. Next we describe the slight modifications necessary to use splatting on bcc grids. Here we need to address issues of interpolation. In order to incorporate shading in our rendering algorithm we describe two methods for estimating gradients on grid points of a bcc lattice.

3.1 Storage scheme

For the sake of simplicity and clarity of figures, we first present our storage scheme in 2D, then extend it to 3D.

In 2D the optimal sampling pattern is hexagonal sampling. Hexagonal sampling as described by Eq. 3, results in rather awkward indexing as we still want to sample a rectangular area. The meaning of the matrix in Eq. 3 is to shift the j^{th} row by the amount $\frac{1}{2}T_1 j$. Since this holds for infinite long rows, the result would be the same to shift the same row by $\frac{1}{2}T_1(j-2) = \frac{1}{2}T_1 j - T_1$. Extending this idea and since we actually like to describe a finite, rectangular area, we shift only rows with odd index which is achieved by the following matrix:

$$V_{hex2D} = \begin{pmatrix} T_1 & \frac{1}{2}T_1(j \bmod 2) \\ 0 & T_2 \end{pmatrix} \qquad \text{for } T_2 = \frac{\sqrt{3}}{2}T_1 \quad (20)$$

The effect is illustrated in Fig. 5. On the left, the result of applying Eq. 3 can be seen whereas the effect of applying Eq. 20 is depicted on the right.

The same problem exists in 3D. However, the solution is as simple as in 2D. The following matrix

$$V_{bcc} = \begin{pmatrix} T & 0 & \frac{1}{2}T(k \bmod 2) \\ 0 & T & \frac{1}{2}T(k \bmod 2) \\ 0 & 0 & \frac{1}{2}T \end{pmatrix} \qquad (21)$$

shifts only planes with odd z-coordinates half a unit in x and y direction. The result is that the slices with even z coordinates make up a 3D Cartesian grid, the slices with odd z coordinates also make up a 3D Cartesian grid which is shifted to the centers of the first grid. Fig. 6 shows a bcc grid with the two inter-penetrating Cartesian grids marked differently. In practice we still store the data in a 3D array with an implicit shift of slices with odd z coordinates.

3.2 Splatting on bcc grids

Adapting Westover‘s splatting algorithm [19] to bcc grids is straightforward. This algorithm gains its power by using spherical reconstruction kernels. These kernels have a spherical extent in the frequency domain. Hence these kernels preserve a spherical region during the reconstruction process. Since the aliased spectra for the hexagonal grid are redistributed so that they do not overlap with the primary spectrum, we can use the existing spherical kernels without any modifications.

Since the data is still organized in a 3D array, we traverse it in a back to front manner. Care has to be taken when traversing in z-direction as planes with odd and even z-coordinates have to be separated. Before applying the transformation matrix we apply the sampling matrix (Eq. 21) to shift the voxels to the correct position.

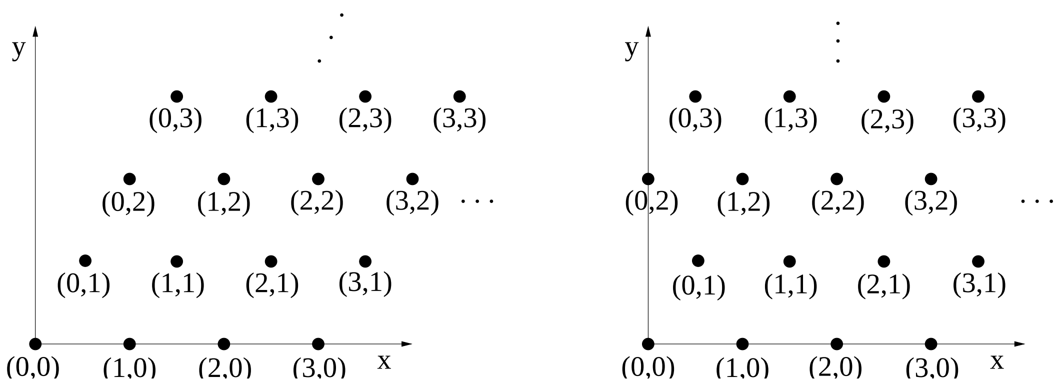


Figure 5: Different indexing schemes. The image on the left corresponds to Eq. 3. The figure on the right corresponds to Eq. 20

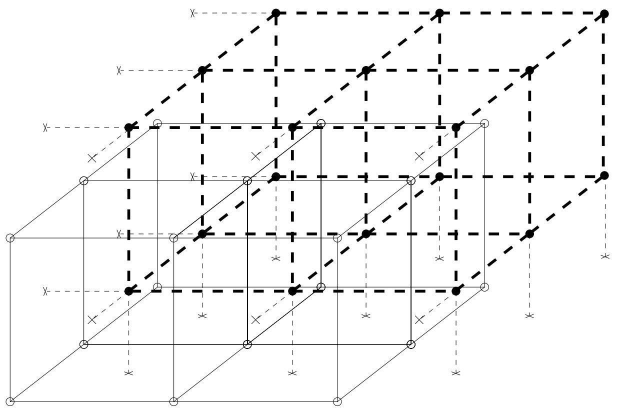


Figure 6: A bcc grid interpreted as two inter-penetrating Cartesian grids.

One more thing has to be changed in existing code: the computation of gradients for shading. For this purpose, we adapted central differences to bcc grids.

3.3 Central Differences on bcc grids

Gradients are rather important in volume visualization. They are most often used for classification and shading. Therefore, we need to be able to compute gradients on bcc grids. The most commonly used method to estimate gradients is the central differences method. There are two ways to adapt this method to bcc grids.

The first idea exploits the fact that we have a Cartesian grid structure in all the slices that are parallel to a major axis direction. Hence partial derivatives in each direction can be computed through standard central differences. However, using our indexing scheme we have to adopt the following equation for computing the central difference in the z direction:

$$f_z[x,y,z] = \frac{1}{2T}(f[x,y,z+2] - f[x,y,z-2]) \tag{22}$$

This method requires exactly as many operations as central differences on Cartesian grids. The conceptual problem with this method is that we do not use the actual closest points in order to estimate the derivatives.

This can be rectified in our second method. For the second method we follow the philosophy that the eight closest points should have the main impact on the reconstructed value. Hence we are computing the average of the central differences at each edge of the cubic cell that the current point is located in (compare Fig. 4). This corresponds to applying an analytic, spherically symmetric, trilinear derivative filter at grid points, resulting in the following formulas for the partial derivatives:

$$\begin{aligned} f_x[x,y,z] &= \frac{1}{4T}\sum_{\substack{i,j\in\{0,1\}\\k\in\{-1,1\}}} h(i) f[\overline{x}-i, \overline{y}-j, z-k] \\ f_y[x,y,z] &= \frac{1}{4T}\sum_{\substack{i,j\in\{0,1\}\\k\in\{-1,1\}}} h(j) f[\overline{x}-i, \overline{y}-j, z-k] \\ f_z[x,y,z] &= \frac{1}{4T}\sum_{\substack{i,j\in\{0,1\}\\k\in\{-1,1\}}} h(k) f[\overline{x}-i, \overline{y}-j, z-k] \end{aligned} \tag{23}$$

with $\overline{m} = m + (z \bmod 2)$ and

$$h(x) = \begin{cases} 1 & \text{if} \quad x = 0 \quad \text{or} \quad x = -1 \\ -1 & \text{if} \quad x = 1 \end{cases} \tag{24}$$

The introduction of $\overline{m}$ and $h(x)$ are due to the properties of our storage scheme.

This method requires 8 operations per partial derivative as opposed to one subtraction per partial derivative for Cartesian grids. However, as we are calculating the gradients in a preprocessing step, this has no major impact on the rendering performance.

4 Acquisition of Optimally Regular Sampled Volume Data

After having settled technical details about using optimally sampled regular volume data, we have to deal with the question where to get data sets sampled on such bcc grids. There are several possibilities.

The first possibility is to sample artificial data sets, given as 3D analytic functions, like a sphere or the Marschner-Lobb function [11], on a bcc grid. Generally, data sets obtained via voxelization [17] can straightforwardly be generated on a bcc grid. This is especially useful for evaluating the applicability of bcc grids as the frequency content of such data can directly be controlled.

Second, there is of course the possibility to resample data sets on Cartesian grids to a bcc grid. Since this resampling step has to be performed only once when generating the new data set, an arbitrarily good reconstruction kernel (e.g., a rather wide windowed sinc) can be used.

The third and most interesting possibility is to take raw data from modalities like CT or MRI and directly generate bcc grid data sets from them. Raw data from tomography data sets (CT, PET, SPECT) is typically given by many 2D or 1D projections. Hence adapting the reconstruction algorithm for bcc grids has the potential of speeding up these typically very costly operations by almost 30%. Likewise image data acquired in the frequency domain (e.g. MRI) could be (re)sampled onto an fcc grid. We could easily acquire the samples in the frequency domain on a face centered cubic grid and use a modified inverse FFT to generate a bcc grid data set. That would allow either faster acquisition times or more accurate images when samples are acquired on a bcc grid.

5 Results

We performed several tests to evaluate the applicability of bcc grids. First, we compared our gradient reconstruction schemes to the commonly used central differences on Cartesian grids. Then, we modified an existing splatter to operate on bcc grids. Finally, as sampling on bcc grids results in a compression of the data, we also compared it to existing compression techniques for volumetric data.

5.1 Gradient estimation

In order to evaluate the quality of the gradient approximation we used three different analytical functions for comparison purposes:

- spherical with linear falloff:

$$f_1(x,y,z) = \sqrt{x^2+y^2+z^2} \tag{25}$$

- Sinc function:

$$f_2(x,y,z) = \frac{\sin(\sqrt{x^2+y^2+z^2})}{\sqrt{x^2+y^2+z^2}} \tag{26}$$

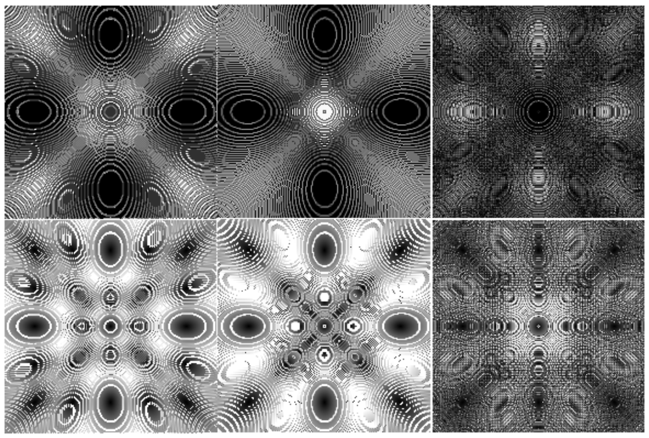
- simplified Marschner-Lobb:

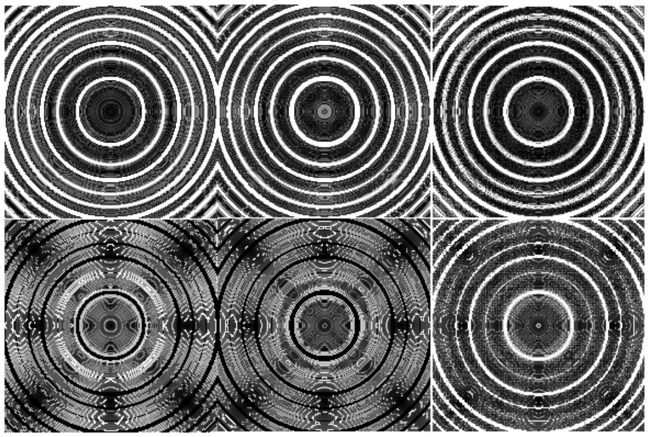
$$f_3(x,y,z) = \sin(x^2+y^2+z^2) \tag{27}$$

We computed the actual function values at the positions defined by the bcc grid so that no errors were introduced during the sampling process. We then applied our two gradient reconstruction schemes and computed the difference of the normals with the analytically computed normals at the sampling grid. We recorded two errors – the error in the magnitude of the normal as well as the angular error in the normal. We then looked at these errors in one slice at a time. Since in our indexing scheme an xy-slice (z is constant) is easy to extract we chose xy slices. Furthermore, we were interested in how well these errors compare to errors introduced by central differences and linear filtering on regular rectilinear grids. Hence we computed the normals as if the original data set was given on a rectangular grid using central differences and linear interpolation.

The results of our experiment can be seen in Fig 7 (a) – (c). The first row shows the relative error in magnitude and the second row shows the angular error. Column one depicts the error of our first gradient reconstruction method (Eq. 22) that is based on central differences at the grid point itself. Column 2 corresponds to method two (Eq. 23), which is the average of all central differences at the of the cube edges surrounding the sampling point. In the last column we computed the linearly interpolated central differences, assuming the data set was given on a regular grid with corresponding dimensions.

Fig 7(a) shows the error images for function f_1. In this image an angular error of 15 degrees and an amplitude error of 30% corresponds to white (255). Fig. 7(b) shows the error images for function


(a)


(b)

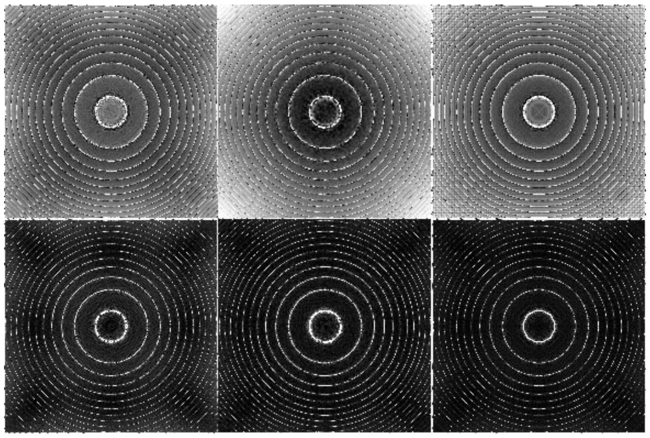

(c)

Figure 7: Difference images of analytically calculated gradients to our gradient estimation schemes (see Sec. 3.3), first two columns, and central differences with linear interpolation, third column, for (a) sphere, (b) Sinc, and (c) simplified Marschner-Lobb function. The top rows show the error in magnitude whereas the bottom rows show the angular error.(a) error in magnitude by 30% and an angular error of 15 degrees corresponds to white, (b) amplitude error of 60% and an angular error of 30 degrees corresponds to white, (c) amplitude error of 10% and an angular error of 5 degrees corresponds to white.

Data set	rectilinear	bcc grid	speedup
uncbrain	1.51	0.8	47%
hipip	0.103	0.059	43%
lobster	0.056	0.043	23%

Table 1: Timings for several different datasets are reported in seconds per frame.

f_2. Here an angular error of 30 degrees and an amplitude error of 60% corresponds to white (255). Finally the results for function f_3 are displayed in Fig. 7(c). Here 5 degrees for the angular error and 10% for the amplitude error correspond to white (255).

From these images we conclude that both our difference methods are quite comparable with central differencing and linear interpolation on regular grids. Hence one need not to worry about quality loss by using bcc grids for volume rendering applications. Furthermore since there are no large differences between the two introduced methods in Section 3.3, we don't find the expensive operations of method 2 justified.

5.2 Splatting

We rendered several different data sets using both a usual Cartesian grid and a bcc grid. All images were generated using the same transfer function and viewing parameters.

Fig. 8 shows images of the Marschner-Lobb data set sampled on a $40 \times 40 \times 40$ Cartesian grid (as described by Marschner and Lobb [11]) on the left respectively a $28 \times 28 \times 56$ bcc grid on the right. This data set is quite demanding for a straightforward splatter and there are some visible differences in the results. The image generated from the bcc grid is rather blurred whereas the image from the Cartesian grid exhibits strong artifacts, especially in diagonal directions. The same phenomenon, but less obvious, can be observed in Fig. 1, which depicts images generated from CT scans of a lobster and a tooth.

The data sets that we used for rendering the images in Color Plate 1 were produced using a high-quality interpolation filter. We used the C^3-4EF filter as designed by Möller et al. [13]. In Color Plate 1 we show results of rendering the "neghip" data set as well as the High Potential Iron Protein data set by Louis Noodleman and David Case, of Scripps Clinic, La Jolla, California, as well as the fuel injection data set. Again, a regular Cartesian grid was used on the left and a bcc grid on the right. There are some visible differences in the images. Since we classify different values that represent two different grid positions one cannot expect identical pictures. Hence we see some differences resulting from the problem of pre-classification [20].

We also did some timings which are reported in Table 1. It is interesting to note that the speedup for some data sets were bigger than expected. This could have been caused by the decreased memory caching necessary. For a very small data set (lobster) we saw expected speedups near 30%.

5.3 Compression

Our results indicate that the resampled data have the potential to lead to better compression. We were able to show that our compression ratios for practical data sets are better than what was achieved using the gzip utility. Also, our overall compression ratios were better then previously reported [5]. Table 2 shows compression ratios of various volume data sets. Note that the last two columns give percentages as compared to the original data size indicating the overall compression ratio, which is what we are interested in. However, the compression of synthetic data sets is a rather surprising result and needs to be further investigated.

6 Future Work

We would like to adopt other volume rendering algorithms to body centered cubic grids and see how they perform. As the bcc grid actually consists of two Cartesian grids which are shifted with respect to each other, it should be possible to design a shear-warp algorithm for bcc grids. For the same reason it should be possible to accelerate volume rendering by the use of graphics hardware.

It would also be necessary to come up with a ray-casting algorithm. For this purpose it would be interesting to investigate the effects of interpolating within bcc grids. Therefore, we would like to adopt the filter analysis and design method of Möller et al. [13] to bcc grids. Since each sampling point in a bcc grid has eight nearest neighbors (as opposed to a Cartesian grid), spherically symmetric filters deserve a more thorough investigation.

7 Conclusions

We have presented a sampling scheme for volume data which saves 29.3% samples as compared to Cartesian grids. We assume that the functions we are dealing with are isotropic and band-limited, i.e., their frequency spectra are spheres. Therefore, a sampling pattern can be used in a way such that the replicas in frequency domain (introduced by the sampling process) are packed closely. There is no unique sampling pattern which achieves this. However, a body centered cubic grid results in a close packing in frequency domain and is easy to use. With this sampling pattern we reduce data size and improve rendering rates without loss of quality.

To demonstrate the applicability in volume rendering, we have adopted the splatting algorithm to bcc grids. This requires just a few changes of an existing code and is straightforward to implement. In order to perform classification and shading of the data we developed two gradient reconstruction schemes. Empirical experiments with analytical 3D functions show that these are comparable with central differences commonly used on Cartesian grids. We believe that significant gains can be achieved by using bcc grids in volume visualization and volume graphics in general.

8 Acknowledgments

We thank the Master for his inspirations and guidance. We also would like to thank Ziemowit Trzesicki for his help in providing code for the resampling of the grids as well as Steven Kilthau for letting us adopt his splatting code for our bcc grids.

References

[1] N.W. Ashcroft and N.D. Mermin. *Solid State Physics*. Prentice-Hall, Inc., Holt, Rinehart and Winston, 1st edition, 1976.

[2] G. Borgefors. Distance transformations on hexagonal grids. *Pattern Recognition Letters*, 9:97–105, 1989.

[3] J.H Conway and N.J.A. Sloane. *Sphere Packings, Lattices and Groups*. Springer, 3rd edition, 1998.

[4] D. E. Dudgeon and R. M. Mersereau. *Multidimensional Digital Signal Processing*. Prentice-Hall, Inc., Englewood-Cliffs, NJ, 1st edition, 1984.

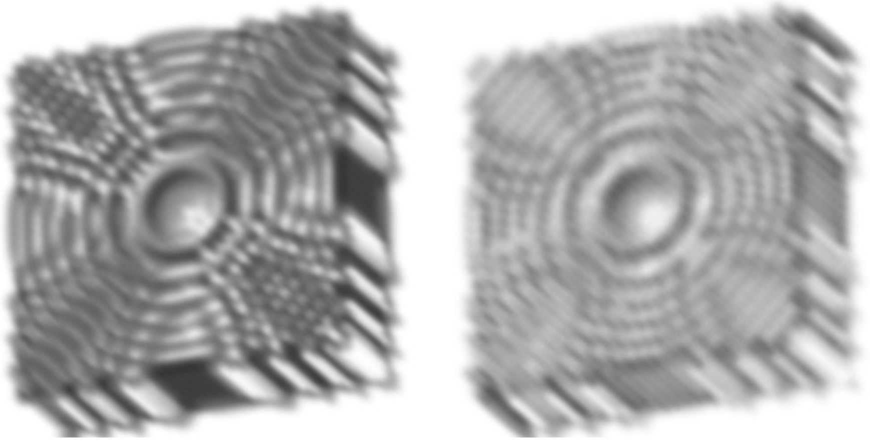


Figure 8: Marschner-Lobb data set rendered by splatting with a Cartesian grid on the left and a bcc grid on the right.

	original	bcc grid	original (gzipped)	bcc (gzipped)	% original	% bcc
uncbrain	9502732	6716017	4547141	2948524	47.8	31.0
nerve	19922955	14021720	8819382	5744443	44.3	28.8
ultrasound	6291467	4422747	4347958	3140634	69.1	49.9
tetrahedron	92410	64024	7028	14456	7.6	15.6
Marschner-Lobb	64001	43913	22823	34861	35.7	54.5

Table 2: Compression ratios of several volume data sets. The last two columns give the percentage as compared to the original data size.

[5] J. Fowler and R. Yagel. Lossless compression of volume data. In Arie Kaufman and Wolfgang Krueger, editors, *Proceedings of the 1994 Symposium on Volume Visualization*, pages 43–50. ACM SIGGRAPH, October 1994. ISBN 0-89791-741-3.

[6] D. Hilbert. Mathematische Probleme. *Nachrichten der Königlichen Gesellschaft der Wissenschaften zu Göttingen, mathematisch-physikalische Klasse*, 3(1):253–297, 1900.

[7] A.G. Jackson. *Handbook of Crystallography*. Springer, 1991.

[8] L. N. Lester and J. Sandor. Computer graphics on a hexagonal grid. *Computers and Graphics*, 8(1):401–409, 1984.

[9] Yong-Kui Liu. The generation of straight lines on hexagonal grids. *Computer Graphics Forum*, 12(1):27–31, March 1993.

[10] M.P. Marder. *Condensed Matter Physics*. John Wiley & Sons, 2000.

[11] S. R. Marschner and R. J. Lobb. An evaluation of reconstruction filters for volume rendering. In R. Daniel Bergeron and Arie E. Kaufman, editors, *Proceedings of the Conference on Visualization*, pages 100–107, Los Alamitos, CA, USA, October 1994. IEEE Computer Society Press.

[12] R.M. Mersereau. The processing of hexagonally sampled two-dimensional signals. *Proceedings of the IEEE*, 67(3):930–946, June 1979.

[13] T. Möller, K. Mueller, Y. Kurzion, R. Machiraju, and R. Yagel. Design of accurate and smooth filters for function and derivative reconstruction. In *1998 Symposium on Volume Visualization*, pages 143–151. ACM SIGGRAPH, October 1998. ISBN 0-8186-9180-8. Held in Research Triangle Park, North Carolina.

[14] N.J.A.Sloane. The sphere packing problem. In *ICM: Proceedings of the International Congress of Mathematicians*, 1998.

[15] A.V. Oppenheim and R.W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Inc., Englewood Cliffs, 2nd edition, 1989.

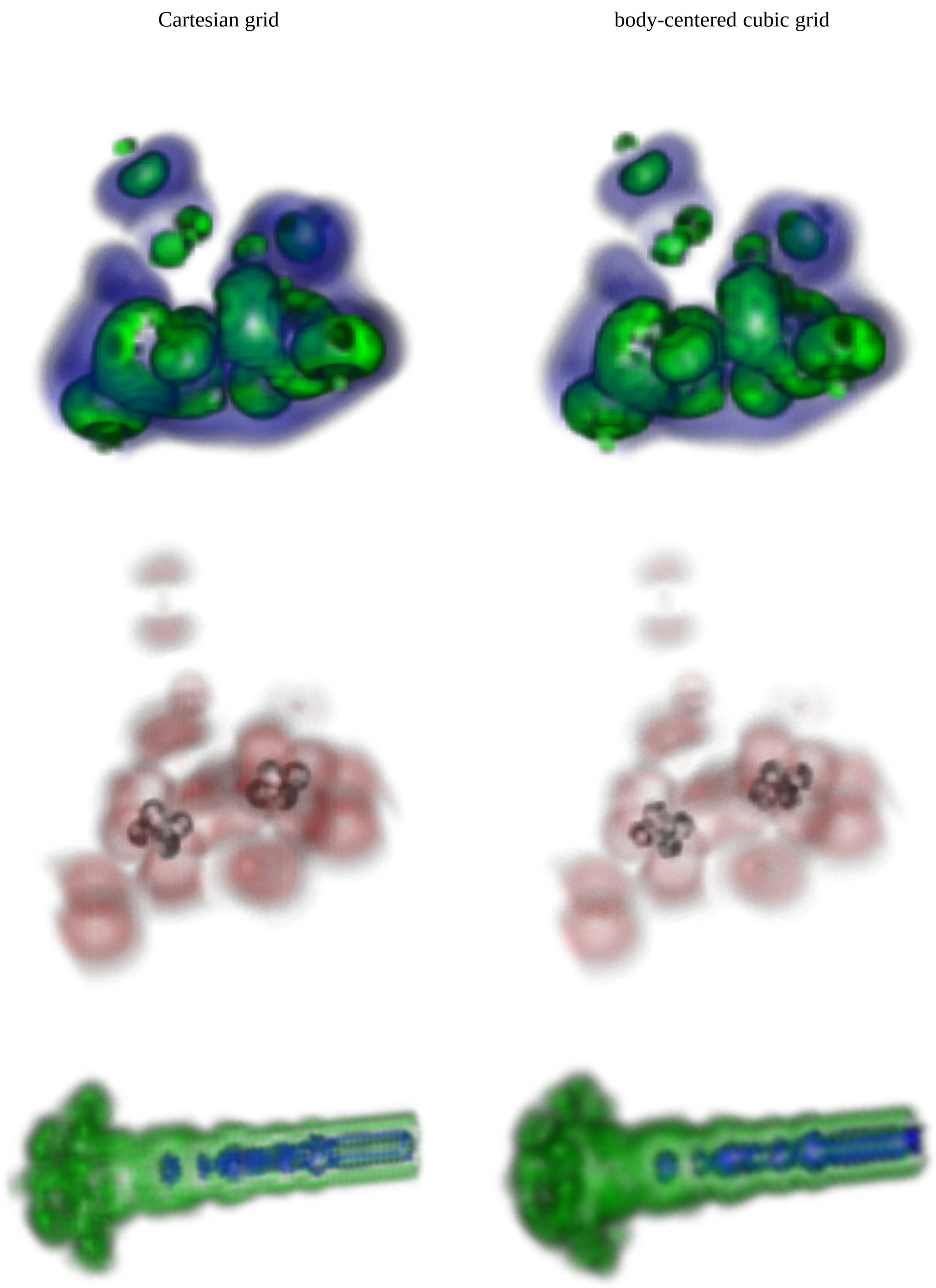
[16] D. P. Petersen and D. Middleton. Sampling and reconstruction of wave-number-limited functions in N-dimensional Euclidean spaces. *Information and Control*, 5(4):279–323, December 1962.

[17] M. Srámek and A. E. Kaufman. Alias-free voxelization of geometric objects. In Hans Hagen, editor, *IEEE Transactions on Visualization and Computer Graphics*, volume 5 (3), pages 251–267. IEEE Computer Society, 1999.

[18] A.F. Wells. *Structural Inorganic Chemistry*. Oxford University Press, 5th edition, 1984.

[19] L. Westover. Footprint evaluation for volume rendering. *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24(4):367–376, August 1990.

[20] Craig M. Wittenbrink, Thomas Malzbender, and Michael E. Goss. Opacity-weighted color interpolation for volume sampling. *1998 Volume Visualization Symposium*, pages 135–142, October 1998. ISBN 0-8186-9180-8.



Color Plate 1: Images generated via splatting on a Cartesian grid on the left respectively a body-centered cubic grid on the right. The body-centered cubic grids require approximately 30% less samples. Small differences are visible, that likely are caused by pre-classification.