Accepted Manuscript

Digital distance functions on three-dimensional grids

Robin Strand, Benedek Nagy, Gunilla Borgefors

PII: S0304-3975(10)00589-X
DOI: 10.1016/j.tcs.2010.10.027
Reference: TCS 8041

To appear in: *Theoretical Computer Science*


This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.
Digital Distance Functions on Three-Dimensional Grids

Robin Strand\textsuperscript{a,}\textsuperscript{*}, Benedek Nagy\textsuperscript{b}, Gunilla Borgefors\textsuperscript{c}

\textsuperscript{a}Centre for Image Analysis, Uppsala University, Sweden
\textsuperscript{b}Department of Computer Science, Faculty of Informatics, University of Debrecen, Hungary
\textsuperscript{c}Centre for Image Analysis, Swedish University of Agricultural Sciences, Sweden

Abstract

In this paper, we examine five different three-dimensional grids suited for image processing. Digital distance functions are defined on the cubic, face-centered cubic, body-centered cubic, honeycomb, and diamond grids. We give the parameters that minimize an error function that favors distance functions with low rotational dependency. We also give an algorithm for computing the distance transform – the tool by which these distance functions can be applied in image processing applications.

Keywords: Digital geometry, Distance functions, Distance transforms, Three-dimensional image processing, Non-standard grids

1. Introduction

Three-dimensional images generated by, e.g., tomographic methods are used as a diagnostic tool in medicine and increasingly in material science and other applications. With higher precision in image acquisition equipment, the amount of data that is used when processing these images is increasing. By using optimal sampling grids, almost 30\% fewer samples are needed to get the same representation/reconstruction quality \cite{1, 2, 3}. Alternatively, better quality can be obtained with the same number of samples. And as the images are computed, rather than captured directly, there is no special reason to use the standard cubic grid.

\textsuperscript{*}Corresponding author

Email addresses: robin@cb.uu.se (Robin Strand), nbenedek@inf.unideb.hu (Benedek Nagy), gunilla@cb.uu.se (Gunilla Borgefors)

Preprint submitted to Theoretical Computer Science August 30, 2010
The possibility of using alternative structures for representing images in two, three, and higher dimensions has inspired researchers to develop image processing methods for non-standard grids [4, 5, 6]. One argument for using non-Cartesian grids is that fewer samples can be used to perfectly reconstruct a band-limited signal when sampled on some of these grids [1, 2, 3, 7, 8]. Given a grid for sampling, the density of the grid in spatial domain is inversely proportional to the density in frequency domain. The denser grid in frequency domain, the larger part of the frequency domain is covered by non-overlapping balls (which is required if aliasing should be avoided). Therefore, one important property when it comes to sampling functions efficiently is related to optimally dense packings. The packing density of a grid is defined as the fraction of the space that is covered by non-overlapping balls of maximal radius centered at the grid points. The densest three-dimensional grid is the face-centered cubic (fcc) grid [9]. The reciprocal grid (the grid in frequency domain) is the body-centered cubic (bcc) grid.

Acquiring images on non-standard grids by tomographic methods is considered in [10, 11, 12]. When reconstructing an object by tomography, each 2D slice of the 3D object that is being imaged is handled separately. If each such slice is reconstructed on a hexagonal grid and the slices are reconstructed without any lateral translation between the slices, the 3D volume image is represented on a honeycomb grid [8, 13]. If lateral translation is applied, the 3D object can be represented by an fcc grid, since the hexagonal grid is embedded in the fcc grid [12].

An important and often used tool in image processing is the distance transform. In a distance transform, each background/object grid point is assigned the minimal distance from the object/background grid point. Some recent papers in which the distance transform is used in applications are found in [14, 15, 16].

In this paper, we will consider the cubic, fcc, bcc, honeycomb, and diamond grids. All these grids, except the diamond grid, are point-lattices, meaning that the grid forms a discrete subgroup of Euclidean space and therefore can be described by a basis. It should also be noted that each of the cubic, fcc, bcc, and diamond grids are unique up to rotation and isotropic scaling. The honeycomb grid consists of layers that are hexagonal grids, and the height between each layer must be defined.

We will analyze distance functions defined as the minimal cost-path using two neighborhood relations on these five grids. With such path-based
distance functions, the distance between two points is calculated by counting the number of (weighted) steps needed to go from one point to the other. We distinguish these digital distance functions from the Euclidean metric which is not discrete in this sense.

To define our digital distance functions, we use weights and neighborhood sequences. This framework includes weighted distances [17, 18, 19, 20] and distances based on neighborhood sequences (ns-distances) [21, 22, 23, 24], which are both generalizations of the well-known city block and chessboard metrics [25]. With weighted ns-distances [26, 27, 28], the rotational dependency of the distance function is potentially low [27, 29].

In this paper, we build on some of our old results, for example [30, 31] presented at the 13th International Workshop on Combinatorial Image Analysis (IWCIA 2009), and give formulas for weighted ns-distances for the five grids considered here. The performance on the different grids is compared by means of the asymptotic rotational dependency. The distance transform, the tool for using the distance functions in image processing, is described in Section 5.

2. Three-Dimensional Grids and Neighborhood Relations

The main motivation for this work is to allow image processing on non-standard, three-dimensional grids. In this section, we will define the five grids considered here and give some basic definitions on neighborhood relations. In the illustrations in this paper, each grid point is associated with a voxel defined as the Voronoi region of the grid at that point. The packing densities of these grids are found in Table 1.
2.1. The Cubic Grid

The cubic grid $\mathbb{Z}^3$ is often used for three-dimensional images. One reason is that there is a natural connection between the cubic grid and the data structure that is often used for storing images in computers.

Let $(x, y, z)$ be the difference between the two points $P$ and $Q$ in $\mathbb{Z}^3$. The points $P$ and $Q$ are

1-neighbors if $|x| + |y| + |z| = 1$

2-neighbors if $\max(|x|, |y|, |z|) = 1$ and $|x| + |y| + |z| = 2$.

See also Figure 1.

Note that the 3-neighbor relations can also be defined in a natural way (i.e., based on Voronoi cells); ns-distances using all three types of neighborhood is described in [23, 32, 33, 34].

2.2. The Face-Centered Cubic Grid

The face-centered cubic (fcc) grid is the densest possible packing in three dimensions. This is the Kepler conjecture stated in 1611 [35], proved to be correct as late as 1998 by Hales [36]. Since the fcc grid is a densest packing, it has been considered for image processing in several papers. See for example [5, 6, 8]. In [37], it is proved that sampling a three-dimensional object (a binary function with smooth boundary) on an fcc grid gives better topology-preserving properties than sampling on a cubic grid with twice as many samples(!).
We use the following definition of the fcc grid: \( F = \{ (x, y, z) \in \mathbb{Z}^3 \text{ and } x + y + z \equiv 0 \pmod{2} \} \).

Let \( (x, y, z) \) be the difference between the two points \( P \) and \( Q \) in \( F \). The points \( P \) and \( Q \) are

1-neighbors if \( \max(|x|, |y|, |z|) = 1 \)
strict 2-neighbors if \( \max(|x|, |y|, |z|) = 2 \) and \( |x| + |y| + |z| = 2 \).

The shape of the voxels and the neighborhood relations are illustrated in Figure 2.

2.3. The Body-Centered Cubic Grid

The body-centered cubic (bcc) grid is the reciprocal of the fcc grid. Therefore, it is optimal for sampling 3D functions according to the multidimensional Shannon sampling theorem [1, 2, 3, 7, 8]. It is defined as \( B = \{ (x, y, z) \in \mathbb{Z}^3 \text{ and } x \equiv y \equiv z \pmod{2} \} \).

Let \( (x, y, z) \) be the difference between the two points \( P \) and \( Q \) in \( B \). The points \( P \) and \( Q \) are

1-neighbors if \( \max(|x|, |y|, |z|) = 1 \)
strict 2-neighbors if \( \max(|x|, |y|, |z|) = 2 \) and \( |x| + |y| + |z| = 2 \).

As can be seen in Figure 3, the voxels in the bcc grid are truncated octahedra and there are eight 1-neighbors and six strict 2-neighbors.
2.4. The Honeycomb Grid

The honeycomb grid \( \mathbb{H} \) is obtained by piling hexagonal grids on top of each other. Therefore, it is natural to generate images on the honeycomb grid if each slice in a tomographic image acquisition method is represented by a hexagonal grid. In [13], digital geometry and computer graphics properties of the honeycomb grid are presented. The grid is spanned by the vectors \((1,0,0),(0,0,1),\left(\frac{1}{2},\frac{\sqrt{3}}{2},0\right)\).

Let \((x,y,z)\) be the difference between the two points \(P\) and \(Q\) in \(\mathbb{D}\). The points \(P\) and \(Q\) are

1-neighbors if \(|(x,y,z)| = 1\)
strict 2-neighbors if \(|(x,y,z)| = \sqrt{2}\).

The neighborhoods are illustrated in Figure 4. Note that \((1,0,0),(0,0,1),\left(\frac{1}{2},\frac{\sqrt{3}}{2},0\right)\) generates a honeycomb grid for any \(m\), where \(m\) is the distance between the hexagonal layers. We use \(m = 1\) here since this has been shown to be optimal for digital (weighted) distance functions in [12].

2.5. The Diamond Grid

The diamond grid, or tetrahedral packing, is found in diamond which is the hardest substance known. It is not a point-lattice, but there is a natural relation between the cubic, fcc, bcc, and the diamond grids [38, 39]. You can say that in the same way that the cubic grid is the 3D grid that corresponds to

Figure 3: (a) A voxel in the bcc grid, (b) voxels corresponding to 1-neighbors, and (c) 2-neighbors.
Figure 4: (a) A voxel in the honeycomb grid, (b) voxels corresponding to 1-neighbors, and (c) 2-neighbors.

the 2D square grid and the fcc and bcc grids correspond to the 2D hexagonal grid, the diamond grid corresponds to the 2D triangular grid.

A definition of the diamond grid is $D = \{(x, y, z) \in \mathbb{Z}^3, x \equiv y \equiv z \pmod{2} \text{ and } x + y + z \in \{0, 1\} \pmod{4}\}$. Which neighboring points a point in the diamond grid has depends on its parity: a point in the diamond grid is even if $x + y + z \equiv 0 \pmod{4}$. Otherwise it is an odd point. Let $(x, y, z)$ be the difference between the two points $P$ and $Q$ in $D$. The points $P$ and $Q$ are

1-neighbors if $\max(|x|, |y|, |z|) = 1$

strict 2-neighbors if $\max(|x|, |y|, |z|) = 2$ and $|x| + |y| + |z| = 4$.

In Figure 5, the neighborhood relations are illustrated.

We note here that the 1-neighborhood plays the role of chemical bonds in diamond (and other materials using this crystal structure). The strict 2-neighborhood has also chemical meaning, since this relation represents the closest same type atoms in GaAs and in some other materials. The 3-neighbor relations can also be defined using Voronoi cells [40]. Moreover, the 4-neighborhood can be defined in natural way by a unit cell [38, 39, 31]. However in this paper we generalize the ns-distances to wns-distances only for the first two neighborhood.

3. Distance Functions

The distance functions we consider here are defined as minimal cost-paths. This is a property which makes them well-suited for efficient digital image
processing algorithms, as we will see in Section 5. Since the weighted ns-distances are generalizations of the weighted distances and ns-distances, the formulas presented here also gives the weighted distance and the ns-distance.

Two points are adjacent if they are 2-neighbors. A neighborhood sequence (ns) $B$ is a sequence $B = (b(i))_{i=1}^{\infty}$, where each $b(i)$ denotes a neighborhood relation in a grid $G$. If $B$ is periodic with period $l$, then we write $B = (b(1), b(2), \ldots, b(l))$. A path in a grid is a sequence of adjacent grid points. A path $P_0, P_1, \ldots, P_n$ is a $B$-path of length $n$ if, for all $i \in \{1, 2, \ldots, n\}$, $P_{i-1}$ and $P_i$ are $b(i)$-neighbors.

**Definition 1.** Given the ns $B$, the ns-distance $d(P_0, P_n; B)$ between the $P_0$ and $P_n$ is the length of (one of) the shortest $B$-path(s) between the points.

Let the real numbers $\alpha$ and $\beta$ (the weights) and a path of length $n$, where exactly $l$ ($l \leq n$) adjacent grid points in the path are strict 2-neighbors, be given. The cost of the path is $(n-l)\alpha + l\beta$. A $B$-path between $P_0$ and $P_n$ is a minimal cost $B$-path if no other $B$-path between the points has lower cost.

**Definition 2.** Given the ns $B$ and the weights $\alpha, \beta$, the weighted ns-distance $d_{\alpha,\beta}(P_0, P_n; B)$ is the cost of a minimal cost $B$-path(s) between the points.

The following notation is used:

$$1^k_B = |\{i : b(i) = 1, 1 \leq i \leq k\}| \text{ and } 2^k_B = |\{i : b(i) = 2, 1 \leq i \leq k\}|.$$

In the formulas below, the weights $\alpha, \beta$ are such that $\alpha \leq \beta \leq 2\alpha$. We remark that the weighted distance is obtained when $B = (2)$ and the ns-
distance is obtained when $\alpha = \beta = 1$. The formulas are valid for any two points $P, Q$ in the grid such that $x \geq y \geq z \geq 0$, where $(x, y, z) = Q - P$.

3.1. The Cubic Grid

Weighted ns-distances in $\mathbb{Z}^3$ were examined in [28]. In [8, 34, 41, 33], formulas for ns-distances (with unit weights) were presented for the cubic grid $\mathbb{Z}^3$. We will use these results to give a formula for weighted ns-distances in $\mathbb{Z}^3$.

The following formula is a slightly modified version of Corollary 3.5 in [8].

Lemma 1 (ns-distance in $\mathbb{Z}^3$). The ns-distance is given by

$$d(P, Q; B) = \begin{cases} 
\min_k \{ k \geq \max \{ x + y + z - 2^k \} \} & \text{if } x < y + z \\
\min_k \{ k \geq 1 + \max \{ x + y + z - 1 - 2^k \} \} & \text{if } x < y + z \\
\min_k \{ k \geq \max \{ x, x + y + z - 2^k \} \} & \text{if } x \geq y + z.
\end{cases}$$

Theorem 1 (weighted ns-distance in $\mathbb{Z}^3$). The weighted ns-distance is given by

$$d_{\alpha, \beta}(P, Q; B) = (2d(P, Q; B) - d(P, Q; (1))) \alpha + (d(P, Q; (1)) - d(P, Q; B)) \beta.$$ 

Proof. When $x, y, z \geq 0$, $(x, y, z) = a_1(1, 0, 0) + a_2(0, 1, 0) + a_3(0, 0, 1)$ for some $a_1, a_2, a_3$. The 1-distance is $a_1 + a_2 + a_3$. If the ns $B$ allows $b$ 2-steps to reach $(x, y, z)$, then there are $a'_1, a'_2, a'_3, b_1, b_2, b_3$ such that the $B$-distance is given by $d = a'_1 + a'_2 + a'_3 + b_1 + b_2 + b_3$, where $(x, y, z) = a'_1(1, 0, 0) + a'_2(0, 1, 0) + a'_3(0, 0, 1) + b_1(1, 1, 0) + b_2(0, 1, 1) + b_3(1, 0, 1)$, where all coefficients are non-negative and $b \geq b_1 + b_2 + b_3$. We now prove that this corresponds to a shortest $B$-path with minimal number of 2-steps:

Assume that there is a shortest $B$-path with smaller number of 2-steps, then some additional local steps are used in such a path and since all other local steps have at least one negative coordinate value, either the number of steps is larger than $d$ or the number of steps equals $d$ and the number of 2-steps is larger than $b_1 + b_2 + b_3$, which means that the assumption is false.

Since we have a shortest $B$-path with minimal number of 2-steps, for any weights $\alpha, \beta$ such that $\alpha \leq \beta \leq 2\alpha$, the weighted ns-distance is $(\sum a'_i)\alpha + (\sum b_i)\beta$. The formula follows from the identity $d(P, Q; (1)) = \sum a_i = \sum a'_i + 2\sum b_i = d(P, Q; B) + \sum b_i$, where $\sum b_i$ is the number of 2-steps in the shortest $B$-path. 

\[ \Box \]
3.2. The Face- and Body-Centered Cubic Grids

The following theorems were first presented as Theorem 1 and 2, respectively, in [27]. They are also found in [8] and [42].

**Theorem 2** (weighted ns-distance in $F$). The weighted ns-distance is given by

$$d_{\alpha,\beta}(P,Q;B) = \begin{cases} k \cdot \alpha & \text{if } x \leq y + z \\ (2k - x) \cdot \alpha + (x - k) \cdot \beta & \text{otherwise,} \end{cases}$$

where $k = \min \left\{ l \geq \max \left( \frac{x + y + z}{2}, x - 2l_B \right) \right\}$. 

**Theorem 3** (weighted ns-distance in $B$). The weighted ns-distance is given by

$$d_{\alpha,\beta}(P,Q;B) = (2k - x) \cdot \alpha + (x - k) \cdot \beta$$

where $k = \min \left\{ l \geq \max \left( \frac{x + y + z}{2}, x - 2l_B \right) \right\}$.

3.3. The Honeycomb Grid

The following theorem gives the weighed ns-distance between points in the honeycomb grid. It is a slightly modified version of Corollary 3.14 in [8]. To make the formula valid for $x \geq y \geq z \geq 0$, we note that when $\frac{y}{x} > \frac{1}{\sqrt{3}}$ the distance value is obtained by changing the coordinates using a rotation by $\pi/3$ in the $xy$-plane.

**Theorem 4** (weighted ns-distance in $H$). The weighted ns-distance is given by

$$d_{\alpha,\beta}(P,Q;B) = (2k - T - z) \cdot \alpha + (T + z - k) \cdot \beta$$

where

$$k = \min \left\{ l \geq \max (T, z, T + z - 2l_B) \right\}$$

and

$$T = \begin{cases} x + \frac{1}{\sqrt{3}}y & \text{if } 0 \leq \frac{y}{x} \leq \frac{1}{\sqrt{3}} \\ \frac{2}{\sqrt{3}}y & \text{otherwise.} \end{cases}$$
3.4. The Diamond Grid

Distances based on neighborhood sequences (i.e., without weights) are examined in [40, 31, 43]. Since the symmetries are not as obvious as on the other grids, we give formulas that hold for arbitrary \( P, Q \). We still use the notation \( Q - P = (x, y, z) \). The following lemma is presented as Theorem 1 in [40].

**Lemma 2** (ns-distance in \( D \)). If \( P \) and \( Q \) have the same parity, then

\[
\begin{align*}
d(P, Q; B) &= \min \sum_{i=1}^{l} b(i) \geq \left\{ \frac{|x| + |y| + |z|}{2}, |x|, |y|, |z| \right\}.
\end{align*}
\]

If

- \( P \) is even, \( Q \) is odd and \( (x, y, z) \) has odd number of negative values or
- \( P \) is odd, \( Q \) is even and \( (x, y, z) \) has even number of negative values

then

\[
\begin{align*}
d(P, Q; B) &= \min \sum_{i=1}^{l} b(i) \geq \left\{ \frac{|x| + |y| + |z| - 1}{2}, |x|, |y|, |z| \right\}.
\end{align*}
\]

If

- \( P \) is even, \( Q \) is odd and \( (x, y, z) \) has even number of negative values or
- \( P \) is odd, \( Q \) is even and \( (x, y, z) \) has odd number of negative values

then

\[
\begin{align*}
d(P, Q; B) &= \min \sum_{i=1}^{l} b(i) \geq \left\{ \frac{|x| + |y| + |z| + 1}{2}, |x|, |y|, |z| \right\}.
\end{align*}
\]

**Remark 1.** The diamond grid has the following property: Two point \( P \neq Q \) are 2-neighbors iff there is a 1-path of length two between the points.

**Theorem 5** (weighted ns-distance in \( D \)). The weighted ns-distance is given by

\[
d_{\alpha, \beta}(P, Q; B) = (2d(P, Q; B) - d(P, Q; (1)))\alpha + (d(P, Q; (1)) - d(P, Q; B))\beta.
\]
Proof. We prove that, any shortest $B$-path between two points $P$ and $Q$ is also a minimal cost-path with weights $\alpha, \beta$ such that $0 \leq \alpha \leq \beta \leq 2\alpha$: If it is not a minimal cost-path, then there is a path with lower cost (but not shorter) than the given path. This means that the number of strict 2-steps is larger, but by Remark 1, any $B$-path consists if as many strict 2-steps as possible. Therefore, the ns-distance given in Lemma 2 can be used to define also the weighted ns-distance. The formula follows from the identity given in the proof of Theorem 1.

4. Parameter Optimization

Finding optimal parameters for digital distance functions has been considered by many authors using different error functions \[8, 18, 19, 20, 26, 44, 45, 46, 47, 48\]. The goal is to find parameters that give minimal rotational dependency for the digital distance functions. In this way, we can approximate the Euclidean distance without leaving the framework of digital distance functions.

In this section, we use the compactness measure

$$E = \frac{A^3}{V^2} \frac{1}{36\pi} - 1,$$

where $V$ is the volume and $A$ is the area of the boundary of a polyhedron. The polyhedra used here correspond to the asymptotic shapes of the digital balls generated by the distance functions presented in Section 3.

In [49], we show that the compactness measure is close to the asymptotic optimal also for short distances on the fcc and bcc grids. Experiments have shown that this is the case also for the other grids considered here. However, we focus on the asymptotic behavior in this paper.

The asymptotic compactness measure of balls generated by the distance functions are obtained by using continuous versions of the discrete quantity $1_B$ and allowing $\mathbb{R}^3$ as the domain for the distance functions. We replace $\lim_{k \to \infty} \frac{1_B}{k}$ by a parameter, $\gamma$. Note that $\lim_{k \to \infty} \frac{2_B}{k}$ corresponds to $(1 - \gamma)$. The polyhedra corresponding to the asymptotic shape of the digital balls are obtained in this way. The shape of these polyhedra are used in the optimization.

4.1. The Cubic Grid

The formulas in Lemma 1 and Theorem 1 are rewritten using $\gamma t$, $0 \leq \gamma \leq 1$, $t \in \mathbb{R}$ instead of $1_B$. The continuous value $t$ is used instead of the
discrete value \( k \) to have a distance functions defined for any \((x, y, z) \in \mathbb{R}^3\) such that \( x \geq y \geq z \geq 0 \). We get
\[
d = (2t - x - y - z)\alpha + (x + y + z - t)\beta,
\]
where
\[
t = \begin{cases} 
  x + y + z - (1 - \gamma)t & \text{if } x < y + z \\
  \max\{x, x + y + z - (1 - \gamma)t\} & \text{if } x \geq y + z.
\end{cases}
\]
To get the surface patches that bound the polyhedron, we will solve \( d = r \), where \( r \) is some radius.

4.1.1. Case \( i \): \( x < y + z \)

Now \( t = x + y + z - (1 - \gamma)t \), i.e., \( t = \frac{x + y + z}{2 - \gamma} \) which gives
\[
r = \frac{x + y + z}{2 - \gamma}(2\alpha - \beta) + (x + y + z)(\beta - \alpha).
\]

4.1.2. Case \( ii \): \( x \geq y + z \) and \( x \geq x + y + z - (1 - \gamma)t \)

\( t = x \), so \( r = (x - y - z)\alpha + (y + z)\beta. \)

4.1.3. Case \( iii \): \( x \geq y + z \) and \( x \leq x + y + z - (1 - \gamma)t \)

Now \( t = x + y + z - (1 - \gamma)t \), i.e., \( t = \frac{x + y + z}{2 - \gamma} \) which gives \( r = \frac{x + y + z}{2 - \gamma}(2\alpha - \beta) + (x + y + z)(\beta - \alpha) \).

We note that the shape of the polyhedron depends on \( \beta/\alpha \) and \( \gamma \). The formulas above describe planes which bound a polyhedron in the region \( x \geq y \geq z \geq 0 \). By symmetry, the entire polyhedron is described. The vertices are, up to symmetry, given by
\[
\left(\frac{r}{\alpha}(1, 0, 0) \text{ and } \frac{r}{\beta + \alpha\gamma - \beta\gamma}(1, 1 - \gamma, 0). \right.
\]
The optimal values \( \frac{r}{\alpha} \approx 1.307 \) and \( \gamma = 0 \) are obtained by optimizing \( E \) with respect to these two parameters. These values give \( E \approx 0.137 \). Polyhedra for some parameters are shown in Figure 6. The polyhedron for the parameters that minimize \( E \) is shown in Figure 11.

4.2. The fcc and bcc Grids

The optimization for the fcc and bcc grids are found in \[8, 27, 29\]. The shape of polyhedra for some parameters are found in Figure 7 and 8. Optimal values are found in Table 2. The shape of the “optimal” polyhedra are shown in Figure 11.
4.3. The Honeycomb Grid

By replacing $1_B^k$ by $\gamma t$ in Theorem 4 and rewriting the formula, we get

$$d = (2t - T - z) \cdot \alpha + (T + z - t) \cdot \beta,$$
where

$$t = \max(T, z, T + z - (1 - \gamma)t)$$ and

$$T = \begin{cases} 
    x + \frac{y}{\sqrt{3}} & \text{if } 0 \leq \frac{y}{x} \leq \frac{1}{\sqrt{3}} \\
    \frac{x}{2} & \text{otherwise.}
\end{cases}$$

4.3.1. Case i: $T \geq z, T + z - (1 - \gamma)t$

$$r = (T - z)\alpha + z\beta.$$

4.3.2. Case ii: $z \geq T, T + z - (1 - \gamma)t$

$$r = (z - T)\alpha + T\beta.$$

4.3.3. Case iii: $T + z - (1 - \gamma)t \geq T, z$

In this case $t = \frac{T + z}{2 - \gamma}$, which gives

$$r = \left(\left(\frac{2}{2 - \gamma} - 1\right)\alpha + \left(1 - \frac{1}{2 - \gamma}\right)\beta\right)(T + z).$$
Figure 7: Asymptotic shapes of balls in the fcc grid. Left to right: $\gamma = 0, 0.5, 1$. Top to bottom: $\beta/\alpha = 1, 1.5, 2$. 
Figure 8: Asymptotic shapes of balls in the bcc grid. Left to right: $\gamma = 0, 0.5, 1$. Top to bottom: $\beta/\alpha = 1, 1.5, 2$. 
The shape of the polyhedra bounded by the planes corresponding to the formulas calculated for the three cases are shown in Figure 9 for some parameters. The values that minimize $E$ are found in Table 2 and for the corresponding polyhedron, see Figure 11.

4.4. The Diamond Grid

We use the identity $\sum_{i=1}^{k} b(i) = k + 2B$ before $\gamma t$ is used to replace $1_B$ in Lemma 2. Asymptotically, the parity of the points does not matter, so we use the formula for points of the same parity and get

$$d = (2t - t')\alpha + (t' - t)\beta,$$

$$t : t + (1 - \gamma)t = \max \left\{ \frac{|x| + |y| + |z|}{2}, |x|, |y|, |z| \right\}$$

and

$$t' = \max \left\{ \frac{|x| + |y| + |z|}{2}, |x|, |y|, |z| \right\}.$$

This leads to

$$t = \frac{t'}{2 - \gamma}, \text{ so } d = \left( \frac{\gamma}{2 - \gamma} \alpha + \frac{2}{2 - \gamma} \beta \right) t'.$$
Figure 10: Asymptotic shape of balls in the diamond grid for any parameters.

Figure 11: Asymptotic shapes of balls in the (a) cubic, (b) fcc, (c) bcc, (d) honeycomb, and (e) diamond grids with optimal parameters (see text).

Adjusting the parameters results in a scaling of the polyhedron, so any parameters give the optimal value $E \approx 0.349$. By Remark 1, any neighborhood sequence and weights result in the same polyhedron – a cuboctahedron, see Figure 10 and 11. This interesting phenomenon can also be connected to the fact that the diamond grid is not a lattice, and therefore by using only the two closest neighborhood relations we have not enough freedom. Otherwise using three or four neighborhood relations the situation is much more complex, non-symmetric distance function may appear [40, 31].

Table 2: Optimal parameters and the corresponding values of $E$.

<table>
<thead>
<tr>
<th>grid</th>
<th>$\beta/\alpha$</th>
<th>$\gamma$</th>
<th>$E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{Z}^3$</td>
<td>1.307</td>
<td>0</td>
<td>0.137</td>
</tr>
<tr>
<td>$F$</td>
<td>1.486</td>
<td>0.487</td>
<td>0.127</td>
</tr>
<tr>
<td>$B$</td>
<td>1.220</td>
<td>0.453</td>
<td>0.158</td>
</tr>
<tr>
<td>$H$</td>
<td>1.313</td>
<td>0.424</td>
<td>0.148</td>
</tr>
<tr>
<td>$D$</td>
<td>[1, 2]</td>
<td>[0, 1]</td>
<td>0.349</td>
</tr>
</tbody>
</table>
5. Distance Transforms

The distance transform is a mapping from the image domain, a subset of the grid, to the range of the distance function. In a distance transform, each object grid point is given the distance from the closest background grid point. With the distance functions presented here, the distance values correspond to minimal-cost paths. This implies that the centers of maximal balls (the balls in an object that are not completely covered by any other ball in the object) can be computed efficiently and without errors for some digital distance functions \[8, 50, 51\] opposed to when the Euclidean distance is used \[52\]. The original object can be recovered from the set of centers of maximal balls by computing the reverse distance transform. It is straight-forward to compute the reverse distance transform with digital distance functions \[8\]. When the Euclidean distance is used, this computation is more complex \[53\]. Also, when computing the constrained distance transform (the distance transform on non-convex domains, for example, the geodesic distance) using the Euclidean distance, a complex algorithm based on visible points is needed \[54\]. The corresponding algorithm using a path-based approach is simple, fast, and easy to generalize to higher dimensions \[55, 56\]. The algorithm presented in this section can be used to compute the constrained distance transform, but here we restrict the discussion to the unconstrained case.

The image domain is a finite subset of the grid \(G\) denoted \(\mathcal{I}_G\). We call the function \(F : \mathcal{I}_G \rightarrow \mathbb{R}^d\), where \(\mathbb{R}^d\) is the range of the distance function \(d\), an image.

The union of the set of object grid points and the set of background grid points equals the image domain \(\mathcal{I}_G\). We denote the distance transform for path-based distances with \(DT_C\), where the subscript \(C\) indicates that costs are computed.

**Definition 3.** The distance transform \(DT_C\) generated by the distance function \(d_{\alpha,\beta}(\cdot, \cdot; B)\) of an object \(X \subset \mathcal{I}_G\) is the mapping

\[
DT_C : \mathcal{I}_G \rightarrow \mathbb{R}^d \text{ defined by } \\
P \mapsto d_{\alpha,\beta}(X, P; B), \text{ where } \\
d_{\alpha,\beta}(X, P; B) = \min_{Q \in X} \{d(Q, P)\}.
\]

For weighted ns-distances, the size of the neighborhood allowed in each step is determined by the length of the minimal cost-path (not the cost),
so this value is also needed when propagating distance information. We define the auxiliary transform \( DT_L \) that holds the length of the minimal cost path at each point.

**Definition 4.** The transform \( DT_L \) of an object \( X \subset G \) is the mapping

\[
DT_L: \mathcal{I}_G \rightarrow \mathbb{N} \text{ defined by } P \mapsto d_{1,1}(Q,P;B), \text{ where } Q \text{ is such that } d_{\alpha,\beta}(Q,P;B) = d_{\alpha,\beta}(X,P;B).
\]

**Algorithm 1:** Computing \( DT_C \) and \( DT_L \) for weighted ns-distances by wave-front propagation.

**Input:** \( B, \alpha, \beta \), and an object \( X \subset G \).

**Output:** The distance transforms \( DT_C \) and \( DT_L \).

**Initialization:** Set \( DT_C(P) \leftarrow 0 \) for grid points \( P \in X \) and \( DT_C(P) \leftarrow \infty \) for grid points \( P \in X \). Set \( DT_L = DT_C \). For all grid points \( P \in X \) adjacent to \( X \): push \( (P, DT_C(P)) \) to the list \( L \) of ordered pairs sorted by increasing \( DT_C(P) \).

**Notation:** \( \omega_{\overrightarrow{v}} \) is \( \alpha \) if \( \overrightarrow{v} \) corresponds to a 1-step and \( \beta \) if \( \overrightarrow{v} \) corresponds to a 2-step.

**while** \( L \) is not empty **do**

**foreach** \( P \) in \( L \) with smallest \( DT_C(P) \) **do**

Pop \((P, DT_C(P))\) from \( L \):

**foreach** \( Q: Q,P \text{ are } b(DT_C(P) + 1)\)-neighbors **do**

if \( DT_C(Q) > DT_C(P) + \omega_{\overrightarrow{PQ}} \) **then**

\[
DT_C(Q) \leftarrow DT_C(P) + \omega_{\overrightarrow{PQ}}; \\
DT_L(Q) \leftarrow DT_L(P) + 1; \\
\text{Push } (Q, DT_C(Q)) \text{ to } L;
\]

**end**

**end**

**end**

Algorithm 1 shows how the distance transform can be computed by wave-front propagation. The Algorithm is based on the Dijkstra algorithm, which has time complexity \( O(n \log n) \), where \( n \) is the number of grid points in the image domain. The factor \( n \) is unavoidable, since each grid point has to be
Table 3: Integer approximations of the optimal parameters in Table 2 used to generate the digital balls in Figure 12.

<table>
<thead>
<tr>
<th>Grid</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$B$</th>
<th>Number of Voxels</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{Z}^3$</td>
<td>10</td>
<td>13</td>
<td>(2)</td>
<td>28533</td>
</tr>
<tr>
<td>$F$</td>
<td>2</td>
<td>3</td>
<td>(1,2)</td>
<td>33385</td>
</tr>
<tr>
<td>$B$</td>
<td>5</td>
<td>6</td>
<td>(1,2)</td>
<td>26015</td>
</tr>
<tr>
<td>$H$</td>
<td>10</td>
<td>13</td>
<td>(1,2,1,2,2)</td>
<td>29737</td>
</tr>
<tr>
<td>$D$</td>
<td>1</td>
<td>1</td>
<td>(1,2)</td>
<td>23691</td>
</tr>
</tbody>
</table>

visited at least once and the factor $\log n$ is needed to keep the auxiliary data structure, the list $L$ of ordered pairs, sorted. However, with integer weights, the list can be kept sorted in constant time, since only integer distance values are attained. The sorting is then similar to pigeonhole sort. Also, since the wave-front contains a limited number of distance values, this approach does not lead to any significant increase in space complexity.

In [8], it is proved that Algorithm 1 gives correct results on point-lattices. The distance transform $DT_C$ can be computed in linear time without the additional transform $DT_L$ on point-lattices by using a look-up table [8].

5.1. Digital Balls

Here we give the shape of digital balls, i.e., balls defined by the digital distance functions. Algorithm 1 was used to generate the balls in Figure 12, where each grid point in the ball is represented by the corresponding voxel. The radii of the balls are $20\alpha$. In the implementations, we used integer approximations that give good approximations of the optimal, real-valued parameters. The main reason for using integer parameters is that it is easy and efficient to use integers in the algorithms. The parameters that were used are shown in Table 3.

6. Conclusion and Future Work

With the compactness measure used here, the fcc grid has the lowest rotational dependency. The cubic grid has the second lowest value of the error function. Note that the optimal value for the cubic grid is attained for weighted distances, i.e., the neighborhood sequence does not add anything to the performance. Since the vectors corresponding to 2-neighbors in the cubic grid are collinear with the vectors corresponding to the 2-neighbors on
Figure 12: Digital balls with radius $20\alpha$ (= 20 1-steps) in the (a) cubic, (b) fcc, (c) bcc, (d) honeycomb, and (e) diamond grids with the parameters in Table 3.
the fcc grid, the asymptotic shape of the digital balls are equal for these grids when the weighted distances is considered.

In our future work, we plan to apply other error functions to see if it is true that the fcc grid has lowest rotational dependency in general. The shape of the polyhedron in Figure 11(d) seems to be a good approximation of the Euclidean sphere, so intuitively we believe that the honeycomb grid might be the best choice in some aspects. Also, in cases where the resolution is lower in the z-direction, which is often the case in CT images of living patients, the honeycomb grid handles this problem as easily as the cubic grid. The “height” of the voxels that is optimal for sampling is computed in [57] and some preliminary results on the optimal “height” for digital distance functions have been derived, but are not yet published.

By using a larger number of neighbors, the rotational dependency can be reduced. As we mentioned earlier, it is natural to include three neighborhoods for the cubic and diamond grids, since each voxel meet voxels from three neighborhoods in these grids. In [38, 39], we used four neighborhood relations to define ns-distances (i.e., when $\alpha = \beta = 1$) on the diamond grid.

The algorithm for computing the (constrained) distance transform given in Section 5 gives a powerful tool for most fields of applications in image processing on these three-dimensional grids. Traditional application areas include matching [58], mathematical morphology operators [59], medial representations and skeletonization [8], etc. Recent examples of applications in which the distance transform is used are template matching [14], estimation of blood vessel width [15], and navigation in a 3D environment [16]. Digital distance functions are commonly used in these applications since the digital distance functions are efficient and the algorithms can be based on the well-known wave-propagation technique [55, 56]. This is not the case for the Euclidean distance [54].

It is today a standard procedure to use three-dimensional images in, e.g., medical diagnosis. We believe that the interest in non-Cartesian, three-dimensional grids will increase, since three-dimensional images are becoming more common also in non-medical applications. Handling three-dimensional images is more efficient with grids that are optimal for representing these images.
References


Workshop on Combinatorial Image Analysis (IWCIA 2009), Springer-Verlag, 2009, pp. 109–121.


