UPPSALA
UNIVERSITET

IT 11 082
Examensarbete 15 hp
November 2011

# Digital Distance Functions Defined by Sequences of Weights

Alexander Denev

Institutionen för informationsteknologi
*Department of Information Technology*

UPPSALA
UNIVERSITET

**Teknisk- naturvetenskaplig fakultet**
**UTH-enheten**

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

# Abstract

## Digital Distance Functions Defined by Sequences of Weights

*Alexander Denev*

In this paper, digital distance functions using sequences of weights are studied and used to approximate the Euclidean distance. Sequences of weights that guarantee a low maximum absolute error for path lengths of up to 10000 are calculated.
A necessary condition and a sufficient condition for metricity of this kind of distance function are established.

Handledare: Robin Strand
Ämnesgranskare: Gunilla Borgefors
Examinator: Anders Jansson
IT 11 082
Tryckt av: Reprocentralen ITC

# Contents

# 1 Introduction

A distance function is a function that assigns a distance to any pair of points in some space. A distance transform (DT) is defined as a mapping where each background grid point is assigned the distance (defined by some chosen distance function) to the closest object grid point. A special case is the constrained distance transform (cDT), where only paths not passing through given obstacle grid points are admissible.

For some applications, it is desired that the DT approximates the Euclidean distance well, or at least has a low rotational dependency. Also, computing the DT should be efficient and fast, and algorithms using only integers are often preferred.

Applications of distance transforms in image processing include:

- In digital morphology, applications include basic operations such as erosion, dilation and skeletonization.
- Image segmentation, e.g., when using watershed segmentation.
- Path finding and visibility analysis.
- Template matching, e.g., chamfer matching.

Distance functions can be classified into path-based and non-path-based, with the Euclidean distance being an example of the latter. Some common distance functions and corresponding distance transforms are presented below.

A (non-weighted) path-based distance function, e.g. city block distance and chessboard distance, only counts the number of steps going to neighboring grid points in some neigborhood. The distance between two grid points is defined to be the number of steps on a minimal-cost path consisting of such steps.

A simple DT that was first published in [Rosenfeld and Pfaltz, 1966] is the city block DT, which uses no weights and 4-connectedness as its neighborhood relation.

The chessboard DT uses 8-connectedness (i.e., steps to all pixels in the 3x3 neighborhood surrounding a pixel are allowed).

A weighted distance function assigns costs (weights) to steps going to neighboring grid points in some neigborhood. The distance between two grid points is defined to be the sum of weights on a minimal-cost path consisting of such steps. In [Borgefors, 1986], weighted distance transforms using 3x3 neighborhoods and 5x5 neighborhoods are considered.

Neighborhood sequence distance functions use a non-fixed neighborhood relation - city block and chessboard steps are mixed along a path. These are sometimes called octagonal distances due to the shape of the disks.

In [Strand, 2007], a generalization of weighted distances and the neighborhood sequence distances is presented; by using a non-fixed neighborhood relation together with (non-unit) weights, the distance function has lower rotational dependency.

Distance functions (and distance transforms) are often characterized by their chamfer masks, which show the cost of the allowed steps from any given pixels. Also, disks (in a general sense - e.g., chessboard distance "disks" actually look like squares) of grid points within a given distance from the origin are often calculated and compared.

In this project, a related but more general class of distance functions was considered. In this kind of distance function, a general sequence of weights along a path is used; the neighborhood relationship is fixed, but the weights are neither unit nor fixed. Also, sequences of weights that approximate the Euclidean distance optimally were computed. Since the distance functions of this class are not necessarily metrics, a necessary condition and a sufficient condition for metricity were established.

The optimality criterion used most when comparing methods of finding good weight sequences was the mean error compared to the Euclidean distance over a $k$-radius chessboard disk for some $k \in \mathbb{N}$, although the maximum error was usually recorded as well. The chessboard disk was chosen because it is the smallest region covering all paths consisting of $k$ steps, which is what we get with a weight sequence of length $k$. Due to symmetry, usually only the first octant of such a chessboard disk was considered, with appropriate weights to compensate for overlap at the boundaries between octants in the 2D square grid. However, the optimality criterion used when finding individual weights (i.e., finding the weight minimizing the error on the relevant column of the first octant) was not always the same - sometimes it was the mean squared error, in order to penalize larger errors more. Higher-order $L_p$ norms, and the maximum norm $L_\infty$, were also tested.

## 2 Basic notions

Consider a path of length $n$ - $p_0, p_1, p_2, ..., p_n$, where $n \geq 1$ and $p_i \in \mathbb{Z}^2$ for all $i$ - and a sequence of ordered pairs of real-valued weights $((\alpha_i, \beta_i))_{i=1..\infty}$. The cost of the path is

$$\sum_{i=1}^{n} X(p_{i-1} - p_i, i), \text{ where } X(v,k) = \begin{cases} \alpha_k & \text{if } |v|_1 = 1 \\ \beta_k & \text{if } |v|_1 = 2 \text{ and } |v|_\infty = 1 \\ \infty & \text{else.} \end{cases}$$

The distance between two points is the cost of a minimal cost path between the points. The distance from a point to itself is defined to be 0.

For example, the costs of the paths in Figure 1 are $\alpha_1 + \beta_2 + \beta_3$ (starting from the lower left) and $\alpha_1 + \alpha_2 + \alpha_3$, respectively.

The distance function is in the general case defined by a sequence $((\alpha_i, \beta_i))$, but in this paper, we restrict the discussion to the case where $\alpha < \beta < 2\alpha$, meaning $\alpha_i < \beta_j < 2\alpha_i, \forall i, j$. Also, we keep $\alpha_i = 1$ for all $i$, except in sections 7 and 11, where it is explicitly stated that we do otherwise. These restrictions guarantee a simpler shape of the shortest paths and make finding optimal weights easier.

Figure 1: Examples of paths

Note that the less restrictive $\alpha_i < \beta_i < 2\alpha_i, \forall i$ already makes the analysis more complicated: Consider a sequence with $\alpha_1 = 1$, $\beta_1 = 1.9$, $\alpha_2 = 0.85$ and $\beta_2 = 1.6$. This example violates the often used premise that the lowest-cost path from the origin to any other point in the first octant ($0 \leq y \leq x$) consists of only $(1, 0)$ and $(1, 1)$ steps; the optimal path from $(0, 0)$ to $(1, 1)$ would consist of a horizontal step and a vertical step rather than a single diagonal step.

# 3 Distance Transform algorithm using sequences of weights

A modification of the three-pass chamfering algorithm was implemented in MATLAB. See Figure 2 for the masks used; scanning is left-to-right, top-to-bottom with the first mask, left-to-right, bottom-to-top with the second mask and bottom-to-top, right-to-left with the third mask. Three passes were needed because of the path dependency; for a discussion of this, see [Strand et al., 2006]. Additionally, the number of steps needs to be kept track of separately - the mask is not necessarily the same for every step in a path; we need to know which $i$ to use. This is not to be done in a separate pass computing the chessboard distance, since the number of steps needs to be updated during the three scans with the general weight sequences; the chessboard DT allows the diagonal steps to be taken in an order that may be sub-optimal for this more general DT.

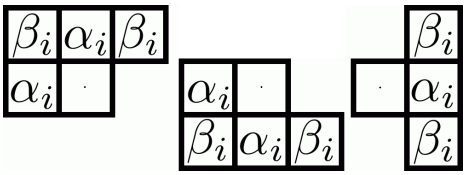


Figure 2: Masks used for DT. From left to right: Masks 1 to 3.

# 4 Finding optimal weights analytically

## 4.1 Starting by matching the Euclidean distance for path length 1

The first method used in the project progresses strictly concentrically outwards. The first step is to match the Euclidean distance for a path length of 1 step (in later steps, we then continue outwards in concentric squares, i.e., chessboard distance disks). This can be done trivially and exactly, since the length of the weight sequence is also 1, and we only need to find $\beta_1$ by solving one linear equation in one unknown (obviously yielding $\beta_1 = \sqrt{2}$). Then, an optimal value for $\beta_2$ was computed analytically on the chessboard disk with radius 2, while keeping the $\beta_1$ previously found to be optimal on the disk with radius 1.

Continuing with the weights in order of increasing index, one can find an optimal $\beta_k$ on a $k$-radius chessboard disk (for some $k \in \mathbb{N}$) given optimal weights found for path lengths $1, 2, ..., k-1$, corresponding to progressively larger concentric chessboard disks, but this was not continued manually. For the error functions used, one can solve the optimality criteria analytically, yielding weights that are linear combinations of square roots of integers (the Euclidean distances on $\mathbb{Z}^2$), with rational numbers as scalars.

We now consider some properties of these lowest-cost paths. Due to symmetry, we sometimes only consider the first octant of the grid ($0 \leq y \leq x$).

Consider the lowest-cost path from $O = (0,0)$ to $(k,i)$, $0 \leq i \leq k$ (i.e. any point in the first octant). Given $\alpha < \beta < 2\alpha$, this lowest-cost path consists of $k-i$ horizontal steps $(1,0)$ with cost 1 and $i$ diagonal steps $(1,1)$ with cost $\sum_{j=1}^{i} \beta'_j$, where $\beta'$ denotes $(\beta_i)_{i=1..k}$ sorted in ascending order. This is because we must take the lowest-cost diagonal steps available (by definition, the distance is the cost of a minimal-cost path; since the number of diagonal steps is fixed, we can only minimize the cost by picking the lowest-cost diagonal steps). Thus, $d(O,(k,i)) = (k-i) + \sum_{j=1}^{i} \beta'_j$. A more detailed discussion is found in Section 4.2.

For an efficient implementation of the algorithm, a useful property is finding where the optimal paths diverge - the path from $(0,0)$ to $(k,k)$ will always end in a diagonal step and the path to $(k,0)$ will always end in a horizontal step. It tuns out that there is exactly one point per column (of the first octant) in which the paths diverge. In order to locate this split, we now consider the optimal paths to the grid points in-between $(0,0)$ to $(k,k)$ in this column, i.e. $(k,1)$ up to $(k,k-1)$, hence we assume $k \geq 2$. Since every step in this lowest-cost path is either $(1,0)$ or $(1,1)$ [Strand, 2007], it follows that this path must pass through either $(k-1,i-1)$ or $(k-1,i)$, and in order to determine which of these two points it passes through, we compare $d(O,(k-1,i-1)) + d(O,(k,i))$ to $d(O,(k-1,i)) + d((k-1,i),(k,i))$, using the convention that we only take the diagonal step if the resulting cost is strictly less than if taking the horizontal step. Hence the last step to $(k,i)$ is diagonal if and only if:

$d(O,(k-1,i-1)) + \beta_k < d(O,(k-1,i)) + 1$

Simplifying this to other equivalent conditions, we get:

$\beta_k < d(O,(k-1,i)) - d(O,(k-1,i-1)) + 1$
$\beta_k < (k - i + \sum_{j=1}^{i} \beta'_j) - (k - (i-1) + \sum_{j=1}^{i-1} \beta'_j) + 1$
Which finally simplifies to:
$\beta_k < \beta'_i$

Note that $\beta'$ here denotes the elements of $(\beta_i)_{i=1..k-1}$, sorted in ascending order; $\beta'_i$ is the $i$-th smallest element of $(\beta_i)_{i=1..k-1}$.

As previously noted, the path to $(k,k)$ always ends with a diagonal step, and the path to $(k,0)$ always ends in a horizontal step. Therefore, and since $\beta'$ is nondecreasing by definition, there is a threshold $j$, $1 \leq j \leq k$ such that the paths from $(0,0)$ to $(k,j)..(k,k)$ all end in a diagonal step and the paths from $(0,0)$ to $(k,0)..(k,j-1)$ all end in a horizontal step.

If the error function used during the optimization to determine the difference between this distance transform and the Euclidean distance makes this feasible, one may solve the error function for an optimal weight $\beta_k$, considering the cases $j = 1..k$ separately and selecting an overall optimal $\beta_k$. An implementation is discussed in the section on numerical solutions.

In Table 1, some results for $k = 2$ are presented. The error function that is minimized is listed in the leftmost column, where MSE denotes mean squared error. The value for $\beta_2$ that minimizes the specified error function given $\beta_1 = \sqrt{2}$ is in the second column. The error values in the two rightmost columns are mean and max of the absolute error over the entire chessboard disk for $k = 2$.

Table 1: Approximating Euclidean distance for $k = 2$

| Error function | Optimal $\beta_2$ | Mean error | Max error |
|---|---|---|---|
| Mean error | $\sqrt{5} - 1 \approx 1.2361$ | 0.0285 | 0.1781 |
| MSE | $\frac{\sqrt{2}+2\sqrt{5}-2}{3} \approx 1.2954$ | 0.0380 | 0.1188 |
| Max error | $\frac{\sqrt{2}+\sqrt{5}-1}{2} \approx 1.3251$ | 0.0428 | 0.0891 |

## 4.2 Matching the Euclidean distance at the boundary of a chessboard disk

The second method used was to match the Euclidean distance at the boundary of a $k$-radius chessboard disk; this task becomes simpler given the restriction $\alpha < \beta < 2\alpha$ stated in the introduction. This yields a sequence of weights sorted in ascending order (denoted $\beta'$). If used directly, in this order, it would yield a very poor approximation of Euclidean distance inside the disk. However, any permutation of $\beta'$ preserves the property of zero error at the boundary, so we select an optimal final weight sequence $\beta$, element by element, by considering concentric chessboard disks, starting from the inside and progressing outwards. The latter part of this procedure is similar to the one described in Section 4.1, with the main difference being that we select the weights from the previously calculated sequence $\beta'$ instead of calculating them now.

Consider the sequence $\beta'$, containing the weights in $(\beta_i)_{i=1..k}$ sorted in ascending order. Given $\alpha < \beta < 2\alpha$, the lowest-cost path from $(0,0)$ to $(k,1)$, $k \geq 1$, consists of $k-1$ horizontal steps $(1,0)$ with cost 1 and 1 diagonal step $(1,1)$ with cost $\beta_x$. This $\beta_x$ must be a minimal element of $(\beta_i)$, hence we choose the first element of the sorted sequence $\beta'$, i.e. $\beta'_1$. We have one linear equation with one unknown, so we can match the Euclidean distance exactly. We then consider the path from $(0,0)$ to $(k,2)$, $k \geq 2$, which consists of $k-2$ horizontal steps (with cost 1) and 2 diagonal steps with cost $\beta'_1$ and $\beta'_2$, respectively. Since we have computed $\beta'_1$, we can again solve one linear equation with one unknown for $\beta'_2$, matching the Euclidean distance to $(k,2)$ (that $\beta'_2 \geq \beta'_1$ indeed holds is proven in Proposition 1). Similarly, one can get a closed-form expression for general $\beta'_i$.

In order to minimize the error inside the $k$-radius boundary of the disk, we need to select the optimal permutation of $(\beta'_i)$ as our weight sequence. One may proceed in a similar manner as in the first approach, by first picking the optimal value for $\beta_1$ on a disk of radius 1, then the optimal value for $\beta_2$ on a disk of radius 2 (given the previously found value for $\beta_1$). In general, finding an optimal $\beta_k$ on a $k$-radius disk given optimal weights found for distances $1, 2, ..., k-1$. Although there is a loss of accuracy for paths with only a few steps, (e.g., we don't have $\sqrt{2}$ available in $\beta'$ to pick for our $\beta_1$), this problem becomes less prominent when we use a longer $\beta'$ sequence, since the gaps between the available elements become smaller.

Some properties of these sorted weight sequences $(\beta'_i)$ are summarized in Proposition 1:

**Proposition 1.** *For all $i$, $k$ such that $1 \leq i \leq k$:*

*(a)* $\beta'_i = 1 + \sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2}$, *where $k$ is the number of steps.*

*(b)* $\sum_{j=1}^{i}(1 + \sqrt{k^2+j^2} - \sqrt{k^2+(j-1)^2}) = \sqrt{k^2+i^2} - (k-i)$

*(c) The arithmetic mean of* $(\beta'_i)_{i=1..k}$ *is* $\sqrt{2}$

*(d) The sequence* $(1 + \sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2})_{i=1..k}$ *is strictly monotonically increasing for all* $k \geq 2$

*(e) As* $k \to \infty$, $\beta'_k \to 1 + \frac{1}{\sqrt{2}}$, *whereas* $\beta'_1 \to 1$

*(f) If we substitute* $y = i/k$, *as* $k \to \infty$, *we get a continuous real-valued function* $f : [0,1] \to [1, 1+\frac{1}{\sqrt{2}}]$, *defined by* $f(y) = 1 + \frac{y}{\sqrt{1+y^2}}$

*(g) The function in (f) is also described by* $f(y) = 1 + \sin(\arctan(y))$

*Proof:*

(a) The cost of the path from $(0,0)$ to $(k,i)$, $k \geq 1$, is $d((0,0),(k,i+1))$. The path consists of $k-i$ horizontal steps $(1,0)$ with cost 1 and $i$ diagonal steps $(1,1)$ with cost $\sum_{j=1}^{i}\beta'_j$ (note that this holds if $\beta'$ is nondecreasing, which it is due to (d); it also holds in the trivial case of $k=1$), hence

(*) $d((0,0),(k,i)) = (k-i) + \sum_{j=1}^{i}\beta'_j$

It follows from (*) that $d((0,0),(k,0)) = k$

We need to match the Euclidean distance between $(0,0)$ and $(k,i)$ exactly:
(**) $d((0,0),(k,i)) = \sqrt{k^2+i^2}$
We will show that (**) $\iff$ $(a)$.
Proof of (**) $\implies$ $(a)$:
From (*), it follows that for any $1 \le i \le k$:
$d((0,0),(k,i)) - d((0,0),(k,i-1)) = \beta'_i + (k-i) - (k-(i-1))$
Which gives:
$\beta'_i = 1 + d((0,0),(k,i)) - d((0,0),(k,i-1))$
Given (**), the above equation implies:
$(a)$ $\beta'_i = 1 + \sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2}$
Proof of $(a)$ $\implies$ (**):
Given (*), (**) is equivalent to:
(***) $\sum_{j=1}^{i} \beta'_j = \sqrt{k^2+i^2} - (k-i)$
From $(b)$ it follows immediately that $(a)$ $\implies$ (***), and since (***) $\iff$ (**) (given (*)), $(a)$ $\implies$ (**).

Thus we have proven (**) $\iff$ $(a)$, and the found $\beta'_i$ weights are the only ones matching the Euclidean distance at $(k,i)$.

(b) $\sum_{j=1}^{i}(1 + \sqrt{k^2+j^2} - \sqrt{k^2+(j-1)^2}) =$
$i + \sum_{j=1}^{i}\sqrt{k^2+j^2} - \sum_{j=0}^{i-1}\sqrt{k^2+j^2} =$
$i + \sqrt{k^2+i^2} - \sqrt{k^2} =$ [since $k > 0$]
$\sqrt{k^2+i^2} - (k-i)$

(c) $\sum_{i=1}^{k} \beta'_i =$ [from $(a)$]
$\sum_{i=1}^{k}(1 + \sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2}) =$
$k + \sum_{i=1}^{k}\sqrt{k^2+i^2} - \sum_{i=0}^{k-1}\sqrt{k^2+i^2} =$ [since $k > 0$]
$k + \sqrt{k^2+k^2} - \sqrt{k^2} = k\sqrt{2}$
Hence the arithmetic mean of $(\beta'_i)_{i=1..k}$ is $\frac{1}{k}\sum_{i=1}^{k}\beta'_i = \sqrt{2}$.

(d) We need to show $\beta'_{i+1} > \beta'_i$ for all $i$, $k$ such that $1 \le i < k$.
From $(a)$, it follows that this is equivalent to:
$1 + \sqrt{k^2+(i+1)^2} - \sqrt{k^2+i^2} > 1 + \sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2}$
$\sqrt{k^2+(i+1)^2} + \sqrt{k^2+(i-1)^2} > 2\sqrt{k^2+i^2}$
Since $1 \le i < k$, this is equivalent to:
$k^2+(i+1)^2 + 2\sqrt{k^2+(i+1)^2}\sqrt{k^2+(i-1)^2} + k^2 + (i-1)^2 > 4k^2 + 4i^2$
$2i^2 + 2 + 2\sqrt{(k^2+(i+1)^2)(k^2+(i-1)^2)} > 2k^2 + 4i^2$
$\sqrt{(k^2+(i+1)^2)(k^2+(i-1)^2)} > k^2 + (i^2-1)$
Since $1 \le i < k$, this is equivalent to:
$k^4 + ((i+1)^2 + (i-1)^2)k^2 + (i+1)^2(i-1)^2 > k^4 + 2k^2(i^2-1) + (i^2-1)^2$
$(2i^2+2)k^2 + (i+1)^2(i-1)^2 > 2k^2(i^2-1) + (i+1)^2(i-1)^2$
$i^2 + 1 > i^2 - 1$

(e) From $(f)$ it follows that as $k \to \infty$, $\beta'_k \to f(1) = 1 + \frac{1}{\sqrt{2}}$ and $\beta'_1 \to f(0) = 1$.
A direct proof based on $(a)$ is omitted.

(f) $\beta'_i = 1 + \sqrt{k^2 + i^2} - \sqrt{k^2 + (i-1)^2} = 1 + \frac{\sqrt{1+(i/k)^2} - \sqrt{1+(i/k-1/k)^2}}{1/k}$
By substituting $y = i/k$:
$f(y) = \lim_{k\to\infty}(1 + \frac{\sqrt{1+y^2} - \sqrt{1+(y-1/k)^2}}{1/k}) = 1 + \frac{d}{dy}\sqrt{1+y^2} =$
$1 + \frac{y}{\sqrt{1+y^2}}$

(g) Let $y = \tan\theta$, $\theta \in [0, \frac{\pi}{4}]$, $y \in [0, 1]$. Note that this is a bijection given these restrictions on domain and codomain, and $\theta = \arctan y$. Then
$f(y) = 1 + \frac{y}{\sqrt{1+y^2}} = 1 + \frac{\tan\theta}{\sqrt{1+\tan^2\theta}} = 1 + \frac{\tan\theta}{\sec\theta} = 1 + \sin\theta = 1 + \sin(\arctan y)$ □

# 5 Finding optimal weights numerically

Using the previously implemented distance transform algorithm (and more efficient, specialized ones), optimal sequences of weights were computed numerically using MATLAB. Both methods mentioned in Section 4 were implemented.

Error functions tested were mean error, mean squared error, higher-order $L_p$ norms, and maximum error, all on the $k$-radius chessboard disk.

Asymptotically faster algorithms (compared to using the distance transform algorithm in a brute force manner, which yielded time complexity $O(k^4)$) were considered, and ones with time complexity $O(k^3)$ were implemented for both methods. We don't need a general DT algorithm for finding the optimal weights - simplified algorithms that compute only the distances from the origin suffice.

For symmetry reasons, the optimization was usually restricted to the first octant, with appropriate weights to compensate for overlap at the boundaries between octants in the 2D square grid. Only one column of this octant needs to be updated when computing a new $\beta_i$ weight. In fact, if visualization of the DT is not needed, we may discard the part of the octant to the left of the currently computed column and the column to its left, while keeping some statistics about the already covered region (such as the maximum error so far).

In the first method, we start by finding $\beta_1$ (keeping the restriction $1 < \beta_i < 2$), then an optimal $\beta_k$ on a $k$-radius disk given optimal weights found for distances $1, 2, ..., k-1$, corresponding to progressively larger concentric chessboard disks.

The search is linear, with adaptive step length. The interval is partitioned into 10 sub-intervals, and the 2 sub-intervals closest to the boundary point with minimal error are chosen for the next iteration, reducing the step length by a factor of 5. The underlying assumption is that the error function is convex. Since this was not proven in the general case, later implementation work instead used the method described in Section 4.1, with the restriction to the $L_2$ norm for performance reasons.

In the second method, we find $(\beta'_i)$, sorted in ascending order, matching the Euclidean distance on the boundary of the largest disk, let us call its radius $k$. In order to get a small error inside the $k$-radius boundary, we make $\beta$ an optimal permutation of $\beta'$ (the observant reader will have noticed that $\beta$ and $\beta'$ are permutations of each other, but not necessarily identical) by picking the

optimal value for $\beta_1$ (by exhaustive linear search), then the optimal value for $\beta_2$ given $\beta_1$ etc., as in the first approach, except that we select weights from $(\beta'_i)$ instead of searching freely inside the open interval $]1, 2[$.

Table 2: Comparison of error function performance, $k = 1000$

| local error function | global mean absolute error | global max absolute error |
|---|---|---|
| mean absolute error | 0.3491 | 1.8433 |
| mean squared error | 0.2675 | 0.8238 |
| $L_4$ | 0.2167 | 0.6563 |
| $L_6$ | 0.2003 | 0.6040 |
| $L_8$ | 0.2064 | 0.6338 |
| $L_{16}$ | 0.2089 | 0.6510 |
| $L_{32}$ | 0.3046 | 1.3727 |
| max (a.k.a. $L_\infty$) | 6.0589 | 15.0204 |

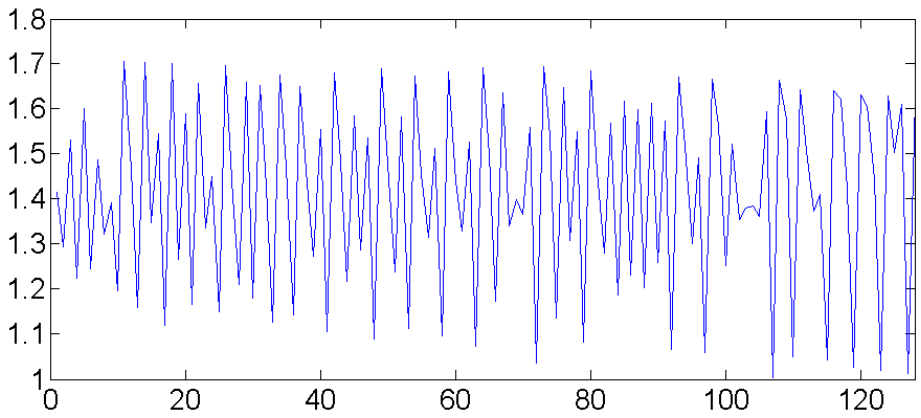


Figure 3: Example $\beta_i$ sequence, $k = 128$, MSE

Results:

- In the first approach (Section 4.1), the average of $(\beta_i)$ apparently converges to $\sqrt{2}$.
- $\beta_2$ is below the average, $\beta_3$ is above the average, with the Fourier spectrum showing "peaks" mostly around $k/2$. See Figure 3 for an example sequence of length 128 (method from Section 4.2, MSE as local error function).

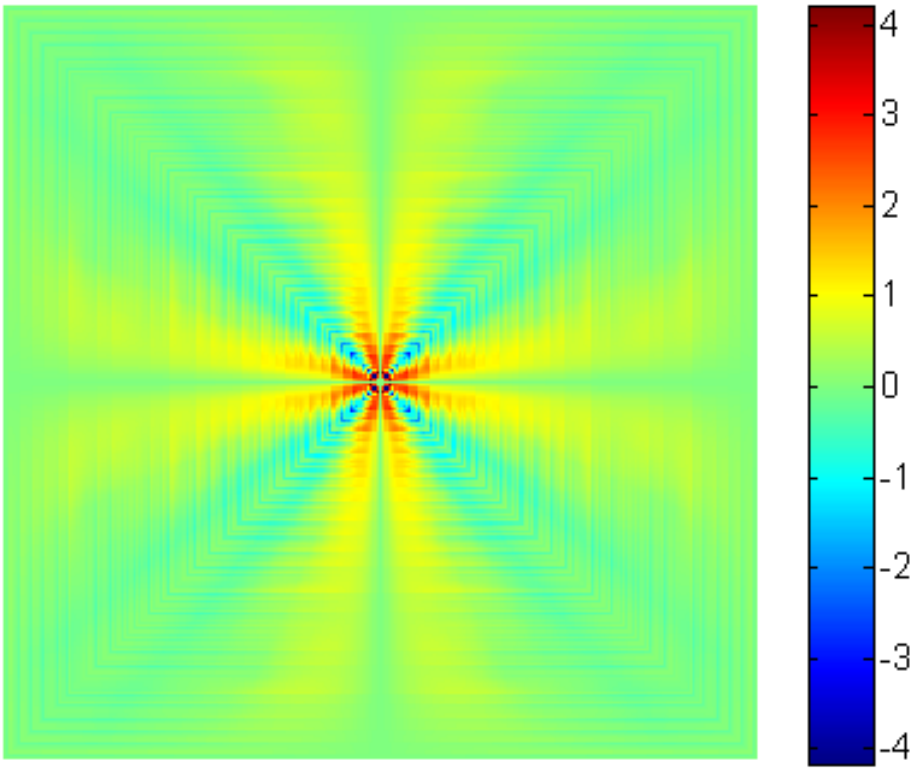


Figure 4: Relative error (in %), $k = 128$, MSE

- As previously noted, we reduce one optimization problem in a high-dimensional space (optimizing the entire weight sequence in order to minimize some error function over the entire disk) to several one-dimensional optimization problems (minimizing the error in one column by picking an optimal single weight). Minimizing a different error function in these smaller problems (compared to the one that would be used in the global problem) sometimes yielded better results. For example, if we want to minimize the maximum error overall, trying to minimize the maximum error for each column yields very poor results - we get apparent convergence of $\beta$ towards a constant. Similarly, using the mean absolute error in the local optimization problems is inferior to using some higher-order norm, at least for sufficiently large $k$. In Table 2, some results are summarized (Algorithm as described in Section 4.2). The $L_6$ norm appears to yield relatively good results. The performance of the maximum norm deteriorates quickly for $k \geq 3$.

- The maximum absolute error stays below one pixel. This has been tested for $k \leq 10000$. The result for $k = 10000$ was a maximum absolute error of 0.6463 and a mean absolute error of 0.2166 (the $L_6$ norm was used).

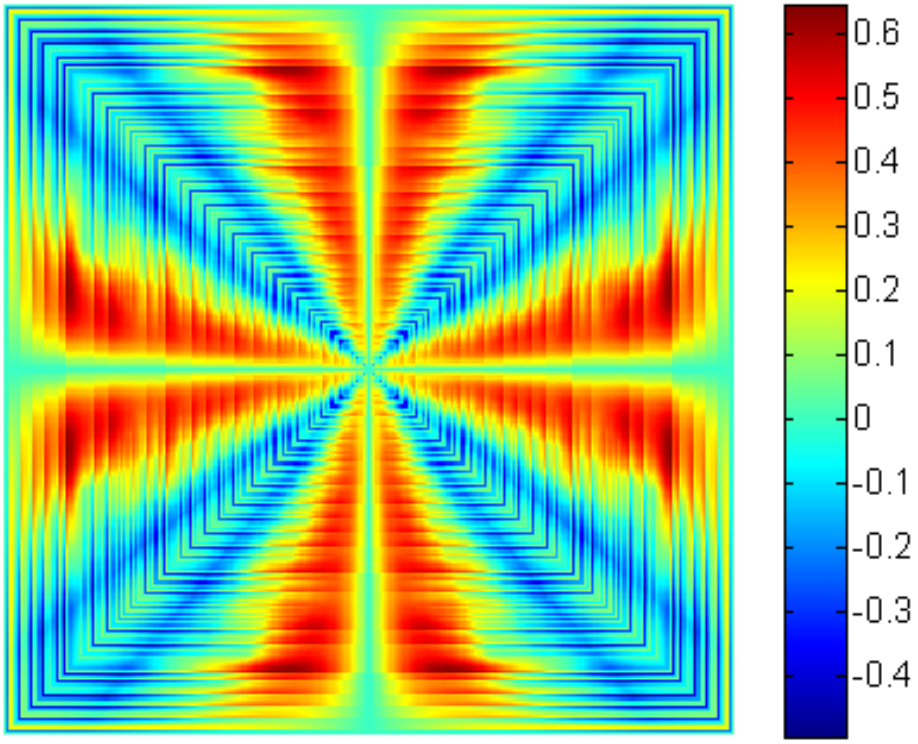


Figure 5: Absolute error, $k = 128$, MSE

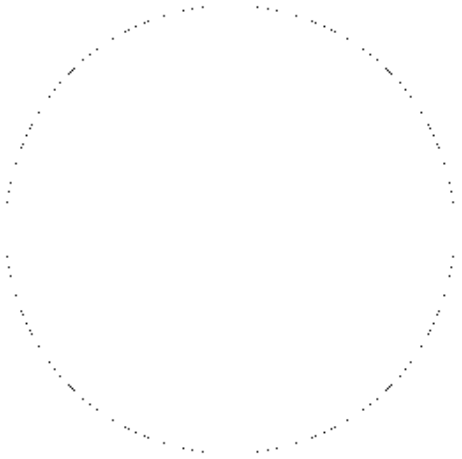

Figure 6: Symmetric difference compared to Euclidean disk, $k = 128$, MSE

This was for the method from Section 4.2, meaning that the error at the boundary is zero (in theory, numerical issues can introduce small errors). For the example sequence in Figure 3, the relative error (compared to Euclidean distance) is shown in Figure 4, the absolute error is shown in Figure 5 and the symmetric difference between the disk and the Euclidean disk is shown in Figure 6.

# 6 Random weight sequences

In order to have a baseline to compare other results with, a random sequence of weights was tested. MATLAB's random number generator was used for this.

The first method was to use $\beta$ from a uniform distribution between 1 and $2\sqrt{2}-1$. These boundaries were chosen to maintain $\alpha < \beta < 2\alpha$ and to keep the mean of the distribution for $\beta$ at $\sqrt{2}$. The uniform distribution was tested early in the project as a baseline that any other method would need to provide superior results to.

The second method was to use a random permutation of the sequence $(\beta'_i)$ described in Section 4, matching the Euclidean distance on the boundary of the chessboard disk, also maintaining a mean $\beta$ of $\sqrt{2}$.

1000 test runs for each method, with $k = 150$, yielded results shown in Table 3.

Table 3: Random $\beta$ weights, $k = 150$

| Method | Mean absolute error | Max absolute error |
|---|---|---|
| Uniform distribution | 1.2932 | 4.3898 |
| Random permutation of $\beta'$ | 0.4904 | 2.1256 |

# 7 Integer weight sequences

Using integer weights is possible, but this was not studied in great detail. One might multiply the end result of the optimization (the weight sequences) with a suitable integer and round the result. Another method that was tested was to use - in each step of the numerical optimization - a step length of $1/n$ for some integer $n$, then multiply all weights by $n$, hence giving $\alpha_i = n$. This indirect approach may add some additional numerical errors. Results for $k = 50$ are shown in Table 4. The error function minimized is mean error. Note that the error is slightly larger for $n = 1000$ than for $n = 100$. This is not because the search for a given $\beta_k$ gets stuck in some poor local minimum; the search was exhaustive for these test runs.

In Table 5, using integer weights is compared to the reference method from Section 4.2, i.e., matching the Euclidean distance exactly on the boundary of a chessboard disk. The error function minimized here is mean squared error.

Table 4: Integer weights, $k = 50$

| $\alpha$ | Mean absolute error | Max absolute error |
|---|---|---|
| 10 | 0.1904 | 0.5465 |
| 100 | 0.1580 | 0.4873 |
| 1000 | 0.1616 | 0.5093 |

Table 5: Integer weights vs. non-integer weights, $k = 250$

| Method | Mean absolute error | Max absolute error |
|---|---|---|
| Integer, $\alpha = 10$ | 0.2655 | 0.8757 |
| Reference (Sec. 4.2) | 0.2286 | 0.6993 |

# 8 Periodic weight sequences

Periodic weight sequences were not studied in great detail. For a periodic weight sequence with period $p$ the resulting shape of the ball is a $8p$-corner polygon, since the resulting shape may be regarded as a morphological convolution of $p$ octagons. The method tested in this project was to use a sequence that matches the Euclidean distance optimally on the border of a $p$-radius chessboard disk, as described previously. In Table 6, some results for periodic sequences are presented, for comparison together with results for the corresponding non-periodic sequence (matching Euclidean distance at the boundary, where $k = 120$). The error function used in the optimization is the mean absolute error, but the maximum absolute error is also presented for comparison purposes.

Table 6: Periodic vs. non-periodic weight sequences, $k = 120$

| Period length | Mean absolute error | Max absolute error |
|---|---|---|
| 2 | 1.1480 | 3.3925 |
| 3 | 0.5142 | 1.5907 |
| 4 | 0.2961 | 0.9126 |
| 5 | 0.2092 | 0.6289 |
| non-periodic | 0.1434 | 0.5106 |

# 9 Approximating the Euclidean circle

Although the main focus of the project was to minimize the mean error compared to the Euclidean distance over a $k$-radius chessboard disk, this was not the

only considered way of approximating the Euclidean distance well. Let us consider the shape of the ball, inspired by the approach in [Hajdu and Hajdu, 2004]. Consider the ideal case of an $8k$-corner regular polygon in $\mathbb{R}^2$ inscribed in a Euclidean circle of radius $k$. The absolute error (maximum distance to the Euclidean circle) in the continuous case is $k(1-\cos\frac{\pi}{8k})$, which converges to 0 as $k \to \infty$. This gives some hope for the error when using our weight sequences in $\mathbb{Z}^2$. Although it is not generally possible to achieve perfectly regular $8k$-corner polygons with radius $k$ in $\mathbb{Z}^2$, a large absolute error in $\mathbb{R}^2$, even for large $k$, would cast doubt upon the feasibility of a good approximation of the Euclidean circle in $\mathbb{Z}^2$.

# 10 Metricity of the distance function

## 10.1 Background

A metric is a generalized distance between elements of a set. If a distance function $d : \mathbb{Z}^2 \times \mathbb{Z}^2 \to \mathbb{R}$ is to be a metric on $\mathbb{Z}^2$, we need the following properties to hold:

**Metricity.** For any points $P, Q, R$ in $\mathbb{Z}^2$,

$d(P,Q) \geq 0$

$d(P,Q) = 0 \iff P = Q$

$d(P,Q) = d(Q,P)$

$d(P,Q) + d(Q,R) \geq d(P,R)$

The first two properties are fulfilled by any distance function of the type we have studied; this follows immediately from the definition in Section 2. The third property (symmetry) is slightly more involved: A minimal-cost path from $Q$ to $P$ can be found by rotating a minimal-cost from $P$ to $Q$ by $\pi$ radians; the cost is the same for both paths. The fourth property (the triangle inequality) does not necessarily hold for all of the distance functions we have studied so far. In fact, many of them are non-metric.

## 10.2 Establishing a necessary condition for metricity

A necessary condition can be found by considering the triangle inequality on diagonal paths, where only $\beta_i$ weights occur.

Consider the points $P = (0,0)$, $Q = (j,j)$ and $R = (k,k)$, $1 \leq j < k$ (for any $j$ and $k$ such that the weight sequence is defined). The cost of the path $\vec{PQ}$ is $d(P,Q) = \sum_{i=1}^{j} \beta_i$, while $d(Q,R) = \sum_{i=1}^{k-j} \beta_i$ and $d(P,R) = \sum_{i=1}^{k} \beta_i$. The triangle inequality requires $d(P,Q) + d(Q,R) \geq d(P,R)$. By subtracting $\sum_{i=1}^{k-j} \beta_i$ from both sides, we get:

$\forall j,k : 1 \leq j < k :$

$\sum_{i=1}^{j} \beta_i \geq \sum_{i=k-j+1}^{k} \beta_i$

In words: For any subsequence $(\beta_1, \beta_2, ..., \beta_k)$, $1 \leq j < k$, the sum of the first $j$ weights must be greater than or equal to the sum of the last $j$ weights (or any $j$ consecutive elements, since it must hold for any $k > 1$). This implies that $\beta_1$ must be a maximal element of $(\beta_i)$, since substituting $j = 1$ in the above expression gives $\beta_1 \geq \beta_k$, $\forall k : k > 1$.

This necessary condition for metricity is violated by some of the otherwise optimal sequences found so far. It is difficult to combine with good approximation of Euclidean distance for both small $k$ and for large $k$. $\beta_1$ should be as close as possible to $\sqrt{2}$, not a maximal element of $\beta$ (which would be above 1.7 in most cases, given $\alpha = 1$).

Note, however, that the necessary condition does not imply that $(\beta_i)$ must be monotonically decreasing or constant; see Figure 7 for a counterexample that is neither. Using this particular weight sequence on the chessboard disk with radius $k = 32$ results in a maximum absolute error of 0.6441, while the mean absolute error is 0.2665. This sequence is constructed recursively from a monotonically decreasing sequence (namely, a $(\beta'_i)$ sequence for matching the Euclidean distance at $k = 32$, sorted in descending order); Figure 8 illustrates the result after only one recursive step - the elements with odd indices are collected in the left half, the ones with even indices to the right. This procedure would then be applied recursively to the two halves, until a stopping condition is reached (a recursive implementation in MATLAB is very short - see Figure 9).

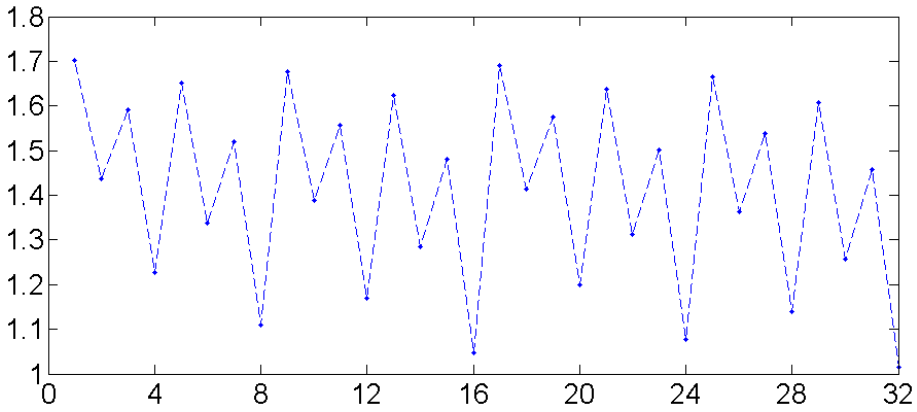


Figure 7: $\beta$ sequence fulfilling a certain necessary condition for metricity

## 10.3 Finding a semimetric

There is also a possibility of finding a semimetric (in the sense of a metric where the triangle inequality is replaced with a $\rho$-relaxed triangle inequality), due to the apparently good approximation of the Euclidean distance. Assume that

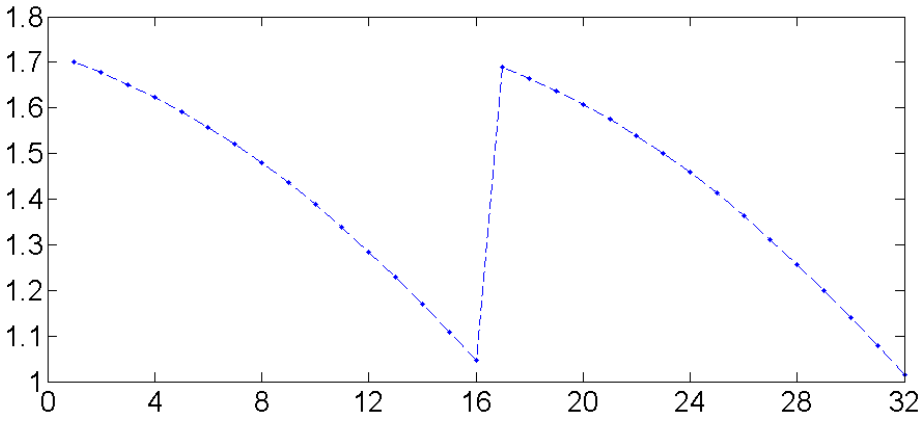


Figure 8: Recursive step

```
function result=recsplit(v,t)
% v: row vector, even length, decreasing
% t: threshold

len=length(v);

if(len>t)
    odd=v(1:2:len-1);    % odd>even if v is decreasing
    even=v(2:2:len);
    result=horzcat(recsplit(odd,t),recsplit(even,t));
else
    result=v;
end;
```

Figure 9: MATLAB source code

there exists some constant $C, C > 1$, such that for any $P, Q \in \mathbb{Z}^2$, $\frac{1}{C} d_E(P,Q) \leq d(P,Q) \leq C d_E(P,Q)$, where $d_E$ is the Euclidean distance. Then, for any three points $P, Q, R \in \mathbb{Z}^2$, $\frac{1}{C} d(P,R) \leq d_E(P,R) \leq d_E(P,Q) + d_E(Q,R) \leq C(d(P,Q) + d(Q,R))$. Putting $\rho = C^2$, we get $d(P,R) \leq \rho(d(P,Q) + d(Q,R))$, which is the $\rho$-relaxed triangle inequality.

## 10.4 Establishing a sufficient condition for metricity

We will show that a monotonically decreasing (i.e., non-increasing) $\beta$ sequence is a sufficient condition for the triangle inequality to hold in full generality, and hence a sufficient condition for metricity.

Also, the type of $\beta$ sequence illustrated in Figure 7 appears promising; we

will comment it briefly in the summary of this report.

We now introduce some notation and lemmata.

Let some arbitrary triangle $ABC$ in $\mathbb{Z}^2$ have side "lengths" (path costs) $|\vec{CB}| = d(C,B) = a$, $|\vec{CA}| = d(C,A) = b$ and $|\vec{BA}| = d(B,A) = c$. We need to show $a + b \geq c$ (obviously, we can apply the same proof method to $b + c \geq a$ and $a + c \geq b$; simply re-labelling the sides is sufficient).

We will use the premise that neither translations nor reflections (of the entire triangle) in the lines $x = 0$, $y = 0$, $y = x$ and $y = -x$ change any side lengths. Therefore, without loss of generality, we may assume $C$ (the corner opposite the side with length $c$) to be in $(0,0)$ and $B$ (the corner opposite the side with length $b$) to be in some $(x_1, y_1)$ in the first octant (hence $x_1 \geq y_1$). $A$ is at some $(x_2, y_2)$, which may be anywhere in $\mathbb{Z}^2$.

For brevity, we define $S(n,k)$ as the sum of the smallest $n$ elements of $(\beta_i)_{i=1..k}$

The side lengths are thus:

$a = x_1 - y_1 + S(y_1, x_1)$

$b = max(|x_2|,|y_2|) - min(|x_2|,|y_2|) + S(min(|x_2|,|y_2|), max(|x_2|,|y_2|))$

$c = max(|x_2 - x_1|,|y_2 - y_1|) - min(|x_2 - x_1|,|y_2 - y_1|) +$
$S(min(|x_2 - x_1|,|y_2 - y_1|), max(|x_2 - x_1|,|y_2 - y_1|))$

**Lemma 1.** $S(n,k+1) \geq S(n,k) - 1$

*Proof:* $S(n,k+1) \geq S(n,k) - 1$ holds since the smallest $n$ elements of $(\beta_i)_{i=1..k+1}$ can differ from the smallest $n$ elements of the $(\beta_i)_{i=1..k}$ in at most one element, thus $S(n,k+1) - (S(n,k))$ equals either 0 or $\beta_i - \beta_j$ for some $i \neq j$, and $\beta_i - \beta_j \geq -1$ due to $1 < \beta < 2$. □

**Lemma 2.** *The distance function $d$ is nondecreasing with increasing city-block distance from the origin:*

*(a)* $d((0,0),(|x|+1,|y|)) \geq d((0,0),(|x|,|y|))$

*(b)* $d((0,0),(|x|,|y|+1)) \geq d((0,0),(|x|,|y|))$

*Proof:*

2(a) Due to symmetry, we need only consider the case $x \geq y$; the other case is handled by a reflection in $y = x$, after which case (b) applies.

$d((0,0),(|x|+1,|y|)) - d((0,0),(|x|,|y|)) = 1 + S(|y|,|x|+1) - S(|y|,|x|) \geq 1 + (S(|y|,|x|) - 1) - S(|y|,|x|) = 0$

Note the use of Lemma 1.

2(b) Due to symmetry, we need only consider the case $|x| \geq |y|$; the other case is handled by a reflection in $y = x$, after which case (a) applies.

$d((0,0),(|x|,|y|+1)) - d((0,0),(|x|,|y|)) = -1 + S(|y|+1,|x|) - S(|y|,|x|) \geq 0$ □

**Lemma 3.**

*(a) If $x_2 > 0$, $c$ does not decrease if we reflect $A$ in the y-axis.*

*(b) If $y_2 > 0$, $c$ does not decrease if we reflect $A$ in the x-axis.*

*Proof:*

3(a) A reflection in the y-axis changes $\vec{BA}$ from $(x_2 - x_1, y_2 - y_1)$ to $((-x_2) - x_1, y_2 - y_1)$. We first prove $|(-x_2) - x_1| \geq |x_2 - x_1|$

In the case $x_2 > x_1$, the inequality becomes:

$x_2 + x_1 \geq x_2 - x_1$

$x_1 \geq 0$

In the case $x_2 \leq x_1$, the inequality becomes:

$x_2 + x_1 \geq x_1 - x_2$

$x_2 \geq 0$

After that, repeated application of Lemma 2(a) suffices to show that $|\vec{BA}| = c$ does not decrease due to the reflection in the y-axis.

The proof of 3(b) is obviously similar. □

**Theorem 1.** *The triangle inequality is fulfilled by any distance function defined by a weight sequence $(\alpha_i, \beta_i)$ such that $\forall i : \alpha_i = 1$, $1 < \beta_i < 2$ and $(\beta_i)$ is monotonically decreasing.*

*Proof:* We will partition the set of all possible triangles by considering the following two cases:

Case I: $|y_2| \leq |x_2|$

Case II: $|y_2| > |x_2|$

Due to Lemma 3, we only need to prove the case $y_2 \leq 0, x_2 \leq 0$ (i.e., when $A$ is in the third quadrant):

If $x_2 > 0$, we reflect $A$ in the $y$-axis. If $y_2 > 0$, we reflect $A$ in the $x$-axis. Let the side length $c$ change to some $c'$ after all required reflections. Lemma 3 ensures that $c' \geq c$, and thus $a + b \geq c' \Rightarrow a + b \geq c$. We will prove $a + b \geq c'$.

However, we can not reduce Case II to Case I in a similar fashion by a final reflection of $A$ in $y = x$; this intuitive idea from Euclidean geometry fails here since $d((0,0),(|x|+1,|y|-1)) \geq d((0,0),(|x|,|y|))$ does not always hold.

For Case I, we actually assume only a more general condition of which a monotonically decreasing $(\beta_i)$ sequence is a special case:

For each $\beta_j$ in any contiguous subsequence $\beta_{k+1}, ..., \beta_{k+x}$, we can assign one distinct $\beta_i$ in $\beta_1, ..., \beta_x$ such that $\beta_i \geq \beta_j$.

Intuitively speaking, it means that any contiguous subsequence is element-wise less than or equal to the contiguous subsequence of the same length that starts at the beginning of the entire sequence - if we sort both subsequences in the same order.

**Case I** $|y_2| \leq |x_2|$:

$a + b \geq c$ is here equivalent to:

$x_1 - y_1 + S(y_1, x_1) + |x_2| - |y_2| + S(|y_2|, |x_2|) \geq |x_2 - x_1| - |y_2 - y_1| + S(|y_2 - y_1|, |x_2 - x_1|)$

*Remark:* Note that we assume $|x_2 - x_1| \geq |y_2 - y_1|$, in this case equivalent to $x_1 + |x_2| \geq y_1 + |y_2|$. This follows from $x_1 \geq y_1$ and $|x_2| \geq |y_2|$. Thus,

geometrically speaking, $\vec{BA}$ is not "steep" - it does not contain any vertical steps.

Thus, the geometric configuration of this case could be characterized as follows:

$A = (x_2, y_2)$ is in the fifth octant.

$B = (x_1, y_1)$ is in the first octant.

$C = (0, 0)$

No side contains any vertical steps.

After some basic arithmetic, we see that the $\alpha$ steps cancel out and that the number of $\beta$ steps is the same on both sides of the inequality:

$S(y_1, x_1) + S(|y_2|, |x_2|) \geq S(|y_2 - y_1|, |x_2 - x_1|)$

Which is equivalent to:

$S(y_1, x_1) + S(|y_2|, |x_2|) \geq S(y_1 + |y_2|, x_1 + |x_2|)$

We will prove this inequality by introducing two intermediate subsequences:

We will find two subsequences (not necessarily contiguous) of $(\beta_i)_{i=1..x_1+|x_2|}$, denoted by $Y_1$ and $Y_2$, such that:

$Y_1$ has $y_1$ elements and $S(y_1, x_1) \geq \Sigma Y_1$

$Y_2$ has $|y_2|$ elements and $S(|y_2|, |x_2|) \geq \Sigma Y_2$

Assume without loss of generality that $|x_2| \geq x_1$ (the reverse case is essentially the same).

We select $Y_1$ by picking the smallest $y_1$ elements of $(\beta_i)_{i=1+|x_2|..x_1+|x_2|}$, which are then elementwise less than or equal to the smallest $y_1$ elements among the first $x_1$ elements of the same sequence (it is true for the entire subsequences of length $x_1$), hence for their respective sums we have $S(y_1, x_1) \geq \Sigma Y_1$.

We select $Y_2$ by picking exactly the same elements as those summed in $S(|y_2|, |x_2|)$ (obviously, $(\beta_i)_{i=1..|x_2|}$ is a subsequence of $(\beta_i)_{i=1..x_1+|x_2|}$ and does not overlap with $(\beta_i)_{i=1+|x_2|..x_1+|x_2|}$) and thus $S(|y_2|, |x_2|) \geq \Sigma Y_2$ holds.

Because of the definition of $S(n, k)$, $\Sigma Y_1 + \Sigma Y_2 \geq S(y_1 + |y_2|, x_1 + |x_2|)$ - any $y_1 + |y_2|$ elements of $(\beta_i)_{i=1..x_1+|x_2|}$ will have a sum that is greater than or equal to the sum of the smallest $y_1 + |y_2|$ elements of $(\beta_i)_{i=1..x_1+|x_2|}$.

Thus, $S(y_1, x_1) + S(|y_2|, |x_2|) \geq \Sigma Y_1 + \Sigma Y_2 \geq S(y_1 + |y_2|, x_1 + |x_2|)$

**Case II** $|y_2| > |x_2|$:

The case where $\vec{BA}$ is "steep" (contains at least one vertical step) can be handled by a reflection of $A$ and $B$ in $y = -x$ (for consistency of notation, we may then swap labels on the corners $A$ and $B$ - we want A in the sixth octant and B in the first octant). If $|x2-x1| < |y2-y1|$ before the reflection in $y = -x$, then $|x2-x1| > |y2-y1|$ will hold after this reflection. Thus we may henceforth assume that $\vec{BA}$ contains no vertical steps.

The geometric configuration of this case could be characterized as follows:

$A = (x_2, y_2)$ is in the sixth octant.

$B = (x_1, y_1)$ is in the first octant.

$C = (0, 0)$

Among the sides, only $\vec{CA}$ contains any vertical steps.

$a + b \geq c$ in this case becomes:

$x_1 - y_1 + S(y_1, x_1) + |y_2| - |x_2| + S(|x_2|, |y_2|) \geq |x_2 - x_1| - |y_2 - y_1| + S(|y_2 - y_1|, |x_2 - x_1|)$

Which is equivalent to:

$2(|y_2| - |x_2|) + S(y_1, x_1) + S(|x_2|, |y_2|) \geq S((|y_2| - |x_2|) + y_1 + |x_2|, x_1 + |x_2|)$

Note that in Case II, unlike in Case I, the $\alpha$ steps do not cancel out and the number of $\beta$ steps is not the same on both sides of the inequality.

We will find three contiguous subsequences of $(\beta_i)_{i=1..x_1+|x_2|}$, denoted by $T$, $Y_1$ and $X_2$, such that:

$T$ has $|y_2| - |x_2|$ elements and $2(|y_2| - |x_2|) \geq \Sigma T$

$Y_1$ has $y_1$ elements and $S(y_1, x_1) \geq \Sigma Y_1$

$X_2$ has $|x_2|$ elements and $S(|x_2|, |y_2|) \geq \Sigma X_2$

We select $T$ by picking the first $|y_2| - |x_2|$ elements of of $(\beta_i)_{i=1..x_1+|x_2|}$. Since $\beta < 2$, it follows that $2(|y_2| - |x_2|) \geq \Sigma T$

We select $Y_1$ by picking the same elements as those summed in $S(y_1, x_1)$ and thus $S(y_1, x_1) \geq \Sigma Y_1$ holds.

We select $X_2$ by picking the smallest $|x_2|$ elements of $(\beta_i)_{i=1..x_1+|x_2|}$ (i.e., the $|x_2|$ last ones, $(\beta_i)_{i=x_1+1..x_1+|x_2|}$), which are then elementwise less than or equal to the smallest $|x_2|$ elements of $(\beta_i)_{i=1..|y_2|}$ (i.e., the last ones), since $|y_2| \leq x_1 + |x_2|$ and $(\beta_i)$ is monotonically decreasing. Hence for their respective sums we have $S(|x_2|, |y_2|) \geq \Sigma X_2$.

Because of the definition of $S(n, k)$, $\Sigma T + \Sigma Y_1 + \Sigma Y_2 \geq S(y_1 + |y_2|, x_1 + |x_2|)$ - any $(|y_2| - |x_2|) + y_1 + |x_2| = y_1 + |y_2|$ elements of $(\beta_i)_{i=1..x_1+|x_2|}$ will have a sum that is greater than or equal to the sum of the smallest $y_1 + |y_2|$ elements of $(\beta_i)_{i=1..x_1+|x_2|}$.

Thus, $2(|y_2| - |x_2|) + S(y_1, x_1) + S(|x_2|, |y_2|) \geq \Sigma T + \Sigma Y_1 + \Sigma X_2 \geq S((|y_2| - |x_2|) + y_1 + |x_2|, x_1 + |x_2|) = S(y_1 + |y_2|, x_1 + |x_2|)$ □

# 11 Looking beyond constant $\alpha$

Many results found in the previous sections can be generalized to weight sequences with non-constant $\alpha$ (we still assume $\alpha < \beta < 2\alpha$). A brief summary is presented below.

Let $\alpha'$ denote $\alpha$ sorted in ascending order, and let $\alpha''$ denote $\alpha$ sorted in descending order.

The cost of the path from $(0, 0)$ to $(k, 1)$ is $\sum_{j=1}^{k-1} \alpha'_j + \beta'_1$. Intuitively speaking, we must replace one horizontal step with one diagonal step, so we replace the highest-cost $\alpha$ with the lowest-cost $\beta$ in order to keep the increase (compared to a strictly horizontal path) as low as possible. In general:

$d(O, (k, i)) = \sum_{j=1}^{k-i} \alpha'_j + \sum_{j=1}^{i} \beta'_j$

We can match the Euclidean distance on the boundary of a $k$-radius ball, using an approach similar to the one used in Section 4.2. We get a closed-form expression for the difference $\beta'_i - \alpha''_i$. The expression found is $\beta'_i - \alpha''_i = \sqrt{k^2 + i^2} - \sqrt{k^2 + (i-1)^2}$. After that, one may find a permutation of the weights that gives a lower error inside the boundary.

For finding an optimal weight sequence - either $(\alpha_i'')$ or $(\beta_i')$ - one possible approach is numerically fitting a line or second-order curve, or a linear interpolation of functions between a line and the function for the difference. The other weight sequence is then determined using the expression for the difference between the two.

In one test of this method, $(\beta_i')$ was manually chosen to be increasing linearly in the interval $[\sqrt{2}-0.15, \sqrt{2}+0.15]$; results are shown in Table 7 together with the corresponding results for constant $\alpha$. The $L_6$ norm was used when minimizing the error. In Figure 10, the absolute error is shown for a test otherwise comparable to the one in Figure 5 (i.e., $k$=128, MSE).

Table 7: Constant vs. varying $\alpha$, $k = 1024$, $L_6$ norm

| $\alpha$ | $\beta$ | Mean absolute error | Max absolute error |
|---|---|---|---|
| 1 | [1.0005, 1.7069] | 0.2046 | 0.5864 |
| [0.8573, 1.2637] | [1.2642, 1.5642] | 0.0530 | 0.1862 |

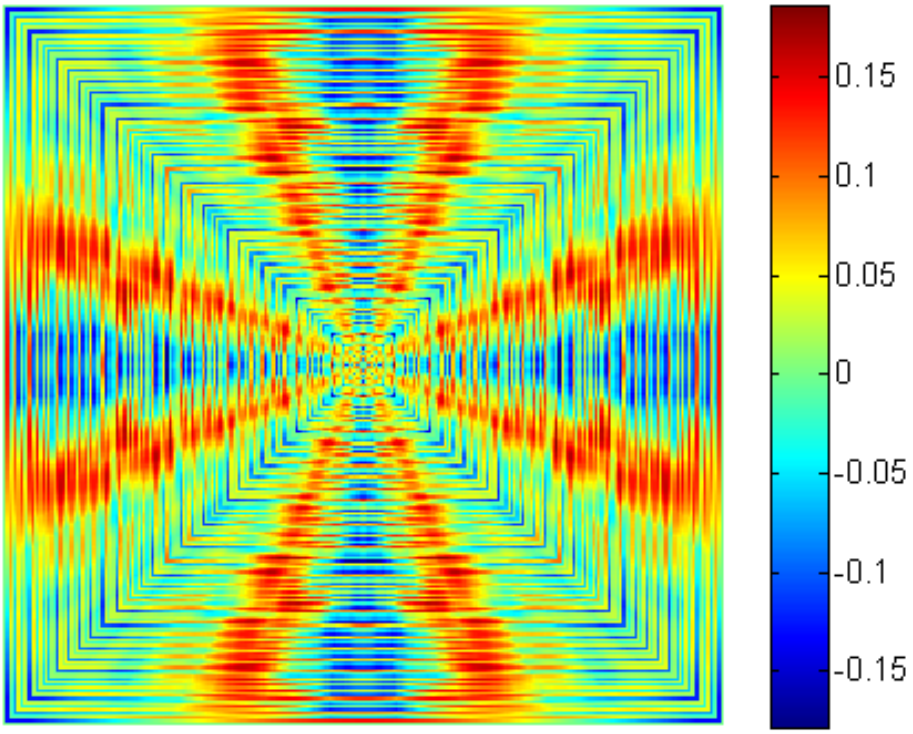


Figure 10: Absolute error (varying $\alpha$, $k = 128$, MSE)

In addition, numerical results were obtained by searching (step length 0.004)

for locally optimal $\alpha$ and $\beta$ independently of each other, without making any assumption about their distribution or difference. Hence this is a generalization of the adaptive step length search described in Section 5. Some results:

- Absolute error halved, approximately.
- Smaller amplitude of weight oscillations.
- "Flat" (uniform) distribution for $\beta$, positive "curvature" (median<mean) for $\alpha$.
- The observed distribution of $\beta'_i - \alpha''_i$ is somewhat similar to the previously mentioned sequence $\sqrt{k^2+i^2} - \sqrt{k^2+(i-1)^2}$, even though there is no matching of the Euclidean distance on the boundary of the ball.

The method described in Section 4.1 is readily generalized to weight sequences with non-constant $\alpha$. Here, the last step on the path from $(0,0)$ to $(k,i)$ is diagonal if and only if:
$\beta_k - \alpha_k < \beta'_i - \alpha''_i$
Compared to the generalization of the method from Section 4.2, it has the advantage of determining both $\alpha$ and $\beta$ optimally. The time complexity is still $O(k^3)$. Due to time constraints, this algorithm was not implemented, however.

# 12 Summary

- The class of distance functions studied can approximate Euclidean distance with a maximum absolute error below one pixel (unit), at least for commonly used resolutions. This has been verified for $k$ up to 10000. It can also match the Euclidean distance exactly at the boundary of a chessboard disk of any given radius.
- One can use a modified raster scan distance transform algorithm with three passes, with time complexity $O(n)$, where $n$ is the number of pixels.
- Integer weights are a fair approximation for medium $k$ (tested for $k$ up to 250), even when using small integers, e.g., $\alpha = 10$.
- Allowing non-constant $\alpha$ reduces the error, and many results for constant $\alpha$ were successfully generalized for this case. A notable exception is the metricity of such distance functions; it was not studied in detail.
- Note that, as previously described, the weight sequences found are not optimized globally, but incrementally on chessboards disks of increasing radius. E.g., the values for $\beta_1$ and $\beta_2$ that minimize the error on a chessboard disk of radius $k = 2$ need not be optimal for $k = 3$. Instead of one optimization problem in a high-dimensional space (i.e., finding a weight sequence minimizing the error in the first octant) we get several one-dimensional optimization problems (i.e., finding a weight minimizing

the error in one column of the first octant, while the preceding weights in the sequence are fixed). The price for this is that the error function to be minimized might need to be changed in these one-dimensional optimization problems. This can be seen when using maximal error as the error function to be minimized - even though the $\beta_3$ found is optimal given the locally optimal $\beta_1$ and $\beta_2$ values, the entire sequence $(\beta_1, \beta_2, \beta_3)$ is not optimal. In fact, trying to directly minimize the maximal error in each step gave poor results for $k \geq 3$. A step towards globally optimized sequences could be to consider $n$-tuples of weights together - e.g., optimizing $\beta_1$ and $\beta_2$ together, then $\beta_3$ and $\beta_4$ together, and so forth.

- Metric distance functions with non-constant weight sequences can be constructed, a simple example being those with monotonically decreasing $\beta$ weights. The type of $\beta$ sequence illustrated in Figure 7 appears to approximate Euclidean distance fairly well, and it looks to be a promising candidate for a more useful metric distance function. The proof of Case I in the proof of Theorem 1 is more general since it is a remnant of these efforts. However, more about this matter was not included in the report.

# References


[Borgefors, 1986] Borgefors, G. (1986). Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344–371.

[Hajdu and Hajdu, 2004] Hajdu, A. and Hajdu, L. (2004). Approximating the Euclidean distance using non-periodic neighbourhood sequences. *Discrete Mathematics*, 283:101–111.

[Rosenfeld and Pfaltz, 1966] Rosenfeld, A. and Pfaltz, J. L. (1966). Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494.

[Strand, 2007] Strand, R. (2007). Weighted distances based on neighbourhood sequences. *Pattern Recognition Letters*, 28(15):2029–2036.

[Strand et al., 2006] Strand, R., Nagy, B., Fouard, C., and Borgefors, G. (2006). Generating distance maps with neighbourhood sequences. In *Proceedings of* $13^{th}$ *International Conference on Discrete Geometry for Computer Imagery (DGCI 2006), Szeged, Hungary*, volume 4245 of *Lecture Notes in Computer Science*, pages 295–307. Springer-Verlag.