

Minimal Cost-Path for Path-Based Distances

Robin Strand, Filip Malmberg, Stina Svensson
Centre for Image Analysis,
Uppsala University and Swedish University of Agricultural Sciences,
Box 337, SE-75105 Uppsala, Sweden

E-mail: {robin,filip,stina}@cb.uu.se

Abstract

Distance functions defined by the minimal cost-path using weights and neighbourhood sequences (n.s.) are considered for the constrained distance transform (CDT). The CDT is then used to find one minimal cost-path between two points. The behaviour of some path-based distance functions is analyzed and a new error function is introduced. It is concluded that the weighted n.s.-distance with two weights (3×3 neighbourhood) and the weighted distance with three weights (5×5 neighbourhood) have similar properties in terms of minimal cost-path computation, while the former is more efficient to compute.

1 Introduction

The weighted distance between two pixels is defined as the cost of the minimal cost-path between the pixels in which each local step in the path is assigned a cost dependent on the local neighbourhood relation the step corresponds to. A popular weighted distance is the 3-4-weighted distance, where the local distance (cost) between edge neighbours is 3 and between vertex neighbours 4, [1]. For distance based on neighbourhood sequences (n.s.-distances), both weights are set to 1 and the neighbourhood relation is allowed to vary along the path [10]. Recently, a path-based distance function – the weighted n.s.-distance – that uses both n.s. and weights was introduced [11]. With this distance function, lower rotational dependency is obtained compared to using only weights (weighted distances) or only n.s. (n.s.-distances), still using information only from a small neighbourhood around each pixel. Note that all these path-based distances (weighted distances, n.s.-distances, and weighted n.s.-distances) are generalizations of the city-block and chessboard distances [9].

In this paper, weighted n.s.-distances with two weights (3×3 neighbourhood) [11] and weighted distances with three weights (5×5 neighbourhood) [1] are considered.

The Euclidean distance function is used in many image processing-applications since it has minimal rotational dependency. However, in some applications, distance functions defined by the minimal cost-path between pixels, i.e. path-based distances, is still preferred. One such example is where the actual minimal cost-path between pixels is of interest to identify, e.g. for the *constrained* distance transform (CDT). To compute the CDT, object, source, and obstacles are identified. The constrained distance between two object points is defined as the minimal cost-path between the points, where obstacles are not allowed to be in the path. The CDT is an image in which each object point is labelled with the constrained distance to their respective closest source pixel. The CDT is used for minimal cost-path planning as the minimal cost-path from a pixel in the object to its closest source pixel can easily be found by walking backwards in the direction of the steepest gradient on the CDT.

The CDT based on a path-based distance function can be computed using standard minimal cost-path techniques for weighted graphs resulting in a linear (in the number of object pixels) time algorithm. In [8], the Dijkstra's graph search algorithm is used. A bucket sorting implementation of the Dijkstra's algorithm is used in [13]. This is a difference compared with the case where the CDT is computed using the Euclidean distance function as only the pixels that are visible from the source pixel can be assigned the Euclidean distance in linear time. To assign the distance to the other pixels, a set of pixels defining discrete straight line segments (DSSs) not intersecting the obstacles are found such that the sum of lengths of the segments is equal to the shortest constrained distance. In [2], a 2D-algorithm that runs in $O(nm)$, where n is the number of object pixels and m is the number of obstacle pixels, is presented as well as an approximate solution that runs in $O(n \log(m))$. Another approach to approximate the Euclidean CDT is to solve the Eikonal equation with fast-marching methods, resulting in an $O(n \log(n))$ algorithm, see e.g. the survey in [4].

In this paper, we give some aspects on the matter of finding *one* minimal cost-path between two points. As

indicated above, the CDT can be used for this purpose. We consider only path-based distances for reasons already explained. For the path-based distances, there are sometimes several minimal cost-paths between two points. In [7], this problem is solved by only considering the path where the number of changes of direction is minimal and then selecting the path with the shortest Euclidean length among these. In general, it is of interest to use a path-based distance function that gives a small set of minimal cost-paths, as in such a case any minimal cost-path can be used which results in a fast linear algorithm that is easy to implement. Moreover, it is then likely that the chosen minimal cost-path has a small deviation from the path with the shortest Euclidean length. In this paper, we approach this by putting in relation the set of possible minimal cost-paths with the shape of a ball calculated by the same distance function. We summarise the properties of path-based distance functions with respect to the size of the set of minimal cost-paths and with focus on using information from a neighbourhood that is as small as possible for the CDT computation. The latter is due to, at least, two reasons. The first reason is that if a neighbourhood larger than 3×3 , e.g. a weighted distance with three weights, is used while computing the CDT, we need to add a connectivity check for local steps between pixels that are not immediate neighbours to avoid jumping over narrow obstacles, see [8]. The second is due to the application we have in mind, which is to apply the framework for grey-level images, i.e. finding shortest grey-weighted paths. Shortest grey-weighted paths is of interest, e.g. in interactive image segmentation where the minimal cost-path represents the boundary of an object of interest, [3]. To obtain smooth object boundaries in regions where the weight image has low contrast, it is desirable to use a grey-weighted DT that gives a small set of minimal cost-paths. The computational complexity of the DT must also be kept low, to allow real-time user interaction. Grey-weighting is often done by multiplying the local spatial distance between two pixels by the mean value of their respective grey-level, [5]. When a weighted distance with more than two weights is used, we, analogous to the CDT computation, need to take special care of local steps between pixels that are not immediate neighbours. This is necessary in order to avoid disregarding narrow grey-level ridges or valleys, [6]. For the comparison we make use of a novel error function, which gives a measure on the number of possible minimal cost-paths.

2 Weighted Distances Based on Neighbourhood Sequences with Two Weights

In this section, the distance functions obtained by using a neighbourhood sequence and two weights is de-

finied. For each grid point, a 3×3 neighbourhood is considered.

Two grid points $\mathbf{p}_1 = (x_1, y_1), \mathbf{p}_2 = (x_2, y_2) \in \mathbb{Z}^2$ are ρ -neighbours, $\rho \in \{1, 2\}$, if

$$\begin{aligned} |x_1 - x_2| + |y_1 - y_2| &\leq \rho \quad \text{and} \quad (1) \\ \max\{|x_1 - x_2|, |y_1 - y_2|\} &= 1. \end{aligned}$$

The points $\mathbf{p}_1, \mathbf{p}_2$ are *adjacent* if \mathbf{p}_1 and \mathbf{p}_2 are ρ -neighbours for some ρ . Two ρ -neighbours such that the equality in (1) is fulfilled are called *strict* ρ -neighbours. A n.s. B is a sequence $B = (b(i))_{i=1}^{\infty}$, where each $b(i) \in \{1, 2\}$ denotes a neighbourhood relation in \mathbb{Z}^2 . If B is periodic, i.e. if for some fixed strictly positive $l \in \mathbb{Z}_+$, $b(i) = b(i + l)$ is valid for all $i \in \mathbb{Z}_+$, then we write $B = (b(1), b(2), \dots, b(l))$.

The following notation is used for the number of 1:s and 2:s in the n.s. up to position k .

$$\begin{aligned} \mathbf{1}_B^k &= |\{i : b(i) = 1, 1 \leq i \leq k\}| \quad \text{and} \\ \mathbf{2}_B^k &= |\{i : b(i) = 2, 1 \leq i \leq k\}|. \end{aligned}$$

A *path* $\mathcal{P} = \langle \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n \rangle$ of length n with starting point \mathbf{p}_0 and end point \mathbf{p}_n , is a sequence $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n$ of adjacent grid points. A path is a *B-path* of length n if, for all $i \in \{1, 2, \dots, n\}$, \mathbf{p}_{i-1} and \mathbf{p}_i are $b(i)$ -neighbours.

Definition 2.1 *Given the n.s. B , the n.s.-distance $d(\mathbf{p}_0, \mathbf{p}_n; B)$ between the points \mathbf{p}_0 and \mathbf{p}_n is the cost of (one of) the minimal cost B -path(s) between the points.*

Let the real numbers α and β (the *weights*) and a B -path \mathcal{P} of length n , where exactly l ($l \leq n$) pairs of adjacent grid points in the path are strict 2-neighbours be given. The *cost of the (α, β) -weighted B -path \mathcal{P}* is $(n-l)\alpha + l\beta$. The B -path \mathcal{P} between the points \mathbf{p}_0 and \mathbf{p}_n is a *minimal cost (α, β) -weighted B -path between the points \mathbf{p}_0 and \mathbf{p}_n* if no other (α, β) -weighted B -path between the points has lower cost than the (α, β) -weighted B -path \mathcal{P} .

Definition 2.2 *Given the n.s. B and the weights α, β , the weighted n.s.-distance $d_{\alpha, \beta}(\mathbf{p}_0, \mathbf{p}_n; B)$ is the cost of (one of) the minimal cost (α, β) -weighted B -path(s) between the points.*

We state now a functional form of the distance between two grid points $(0, 0)$ and (x, y) , where $x \geq y \geq 0$. Observe that by translation-invariance and symmetry, the distance between any two grid points is given by the formula presented in Theorem 2.1. Theorem 2.1 is proved in [11].

Theorem 2.1 *Let the n.s. B , the weights α, β such that $\alpha \leq \beta \leq 2\alpha$, and the point (x, y) , where $x \geq y \geq 0$, be given. The weighted n.s.-distance between $\mathbf{0}$ and (x, y) is given by*

$$\begin{aligned} d_{\alpha, \beta}(\mathbf{0}, (x, y); B) &= (2k - x - y) \cdot \alpha + (x + y - k) \cdot \beta, \\ \text{where } k &= \min_k : k \geq x + \max(0, y - \mathbf{2}_B^k). \end{aligned}$$

Now, the distance function defined for \mathbb{Z}^2 in Theorem 2.1 is generalized to \mathbb{R}^2 . For $\mathbf{p} = (x, y) \in \mathbb{R}^2$ such that $x \geq y \geq 0$, the following distance function, see [11], is considered:

$$d_{\alpha,\beta}(\mathbf{0}, \mathbf{p}; \gamma) = (2t - x - y) \cdot \alpha + (x + y - t) \cdot \beta, \\ \text{where } t : t = x + \max(0, y - (1 - \gamma)t), \quad (2)$$

where $x, y, t \in \mathbb{R}$ and $\gamma \in \mathbb{R}$, $0 \leq \gamma \leq 1$ is the fraction of the steps where 2-steps are *not* allowed (so $\mathbf{1}_B^k$ and $\mathbf{2}_B^k$ corresponds (asymptotically) to γt and $(1 - \gamma)t$, respectively). For example, $B = (1, 2)$ and $B = (2, 1)$ are both represented by $\gamma = 0.5$.

3 Weighted Distances with Three Weights

In this section, the distance functions obtained by using three weights is defined. For each grid point, a 5×5 neighbourhood is considered. Note that we here make use of weights only, and no n.s., in the wanted distance function. However, we could express the distance function as using n.s. $B = (3)$ to stress that 3-steps are allowed in each step. The notion of 3-neighbours is therefore introduced.

Two distinct grid points $\mathbf{p}_1 = (x_1, y_1), \mathbf{p}_2 = (x_2, y_2) \in \mathbb{Z}^2$ are *strict* 3-neighbours if

$$|x_1 - x_2| + |y_1 - y_2| = 3 \text{ and} \\ \max\{|x_1 - x_2|, |y_1 - y_2|\} = 2.$$

Hence, the points $\mathbf{p}_1, \mathbf{p}_2$ are 3-neighbours if they are strict 3-neighbours or 2-neighbours. Let the path $\mathcal{P} = \langle \mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n \rangle$ of length n be such that exactly l pairs of grid points are strict 2-neighbours and exactly m pairs of grid points are strict 3-neighbours. Given three weights α, β, ω , the cost of the (α, β, ω) -weighted path \mathcal{P} is $(n - l - m)\alpha + l\beta + m\omega$.

Definition 3.1 *Given the weights α, β, ω , the weighted distance $d_{\alpha,\beta,\omega}(\mathbf{p}_0, \mathbf{p}_n)$ is the cost of (one of) the minimal cost (α, β, ω) -weighted path(s) between the points.*

For weighted distances with three weights, the following formula from [1] is used for points in both \mathbb{Z}^2 and \mathbb{R}^2 .

Theorem 3.1 *Let the (positive) weights α, β, ω such that $\omega \geq 2\alpha$, $2\omega \geq 3\beta$, and $\omega \leq \alpha + \beta$, and the point (x, y) , where $x \geq y \geq 0$, be given. The (α, β, ω) -weighted distance between $\mathbf{0}$ and $\mathbf{p} = (x, y)$ is given by*

$$d_{\alpha,\beta,\omega}(\mathbf{0}, \mathbf{p}) = \begin{cases} x\alpha + y(\omega - 2\alpha) & \text{for } 2y \leq x \\ x(\omega - \beta) + y(2\beta - \omega) & \text{else} \end{cases}$$

Again, we remark that the weighted distance with three weights defined in this section can be seen as a weighted n.s.-distance defined by the neighbourhood sequence $B = (3)$ and the weights (α, β, ω) .

4 Properties of the Minimal Cost-Paths

In this section, the distances defined for \mathbb{R}^2 in Sections 2 and 3 are considered for the analysis. For any two points $\mathbf{p}, \mathbf{q} \in \mathbb{R}^2$, $d(\mathbf{p}, \mathbf{q})$ is used to denote *either* $d_{\alpha,\beta}(\mathbf{p}, \mathbf{q}; \gamma)$ or $d_{\alpha,\beta,\omega}(\mathbf{p}, \mathbf{q})$.

The shape of the balls are determined by its vertices. For a fixed radius r ,

$$\{\mathbf{q} : d(\mathbf{p}, \mathbf{q}) \leq r\} \quad (3)$$

defines a ball of radius r centered in \mathbf{p} , denoted $\mathcal{B}(\mathbf{p}, r)$. Using Eq. (3) together with Eq. (2) and Theorem 3.1, the vertices of $\mathcal{B}(\mathbf{0}, r)$ are found. They are (up to permutation of the coordinates):

- $\left(\pm r \frac{1}{\beta(1-\gamma)+\alpha\gamma}, \pm r \frac{1-\gamma}{\beta(1-\gamma)+\alpha\gamma}\right)$ and $(\pm \frac{r}{\alpha}, 0)$ for weighted n.s.-distance using two weights (3×3 -neighbourhood).
- $(\pm \frac{r}{\alpha}, 0)$, $(\pm \frac{2r}{\omega}, \pm \frac{r}{\omega})$, and $(\pm \frac{r}{\beta}, \pm \frac{r}{\beta})$ for weighted distances using three weights (5×5 -neighbourhood).

Balls $\mathcal{B}(\mathbf{0}, 1)$ for some values of (α, β, γ) and (α, β, ω) are shown in Figure 1. We remark that we use the notation *weighted n.s.-distance* when the n.s. has effect on the calculated distance and in other cases *weighted distance*. In the figure, optimal values of $\alpha, \beta, \omega, \gamma$ derived in, e.g. [11, 12], are used.

For a given distance function d and a point \mathbf{p} , consider the set

$$S_{\mathbf{p}} = \{\mathbf{q} : d(\mathbf{0}, \mathbf{q}) + d(\mathbf{q}, \mathbf{p}) = d(\mathbf{0}, \mathbf{p})\}.$$

For the Euclidean distance, the set $S_{\mathbf{p}}$ is the straight line between $\mathbf{0}$ and \mathbf{p} . See Figure 2 for some examples in both \mathbb{R}^2 and \mathbb{Z}^2 . Intuitively, the larger set $S_{\mathbf{p}}$, the larger deviation from the Euclidean distance of the distance function. This reasoning is now formalized in order to compare some different distance functions.

Since \mathbf{p} in $S_{\mathbf{p}}$ is fixed, so is $d(\mathbf{0}, \mathbf{p})$. We denote this distance with r . Consider the set $Q_{\mathbf{p},s}$ defined as the set $S_{\mathbf{p}}$ with fixed value of $d(\mathbf{0}, \mathbf{q}) (= s)$, where $0 \leq s \leq r$. The set $Q_{\mathbf{p},s}$ is

$$Q_{\mathbf{p},s} = \{\mathbf{q} : d(\mathbf{0}, \mathbf{q}) + d(\mathbf{q}, \mathbf{p}) = d(\mathbf{0}, \mathbf{p}) \\ \text{where } d(\mathbf{0}, \mathbf{q}) = s\} \\ = \{\mathcal{B}(\mathbf{0}, s) \cap \mathcal{B}(\mathbf{p}, d(\mathbf{0}, \mathbf{p}) - s)\}.$$

By rewriting $Q_{\mathbf{p},s}$ in this way, we see that the number of minimal cost-paths between two points is closely related to the shape of the balls. The set $Q_{\mathbf{p},s}$ is shown in dark grey for $s = r/2$ in Figure 2.

To find the worst case, i.e. the direction that gives the largest set $Q_{\mathbf{p},s}$, we note that for fixed values of the parameters $(\alpha, \beta, \gamma, \text{ and } \omega)$, the set $Q_{\mathbf{p},s}$ is a line (or

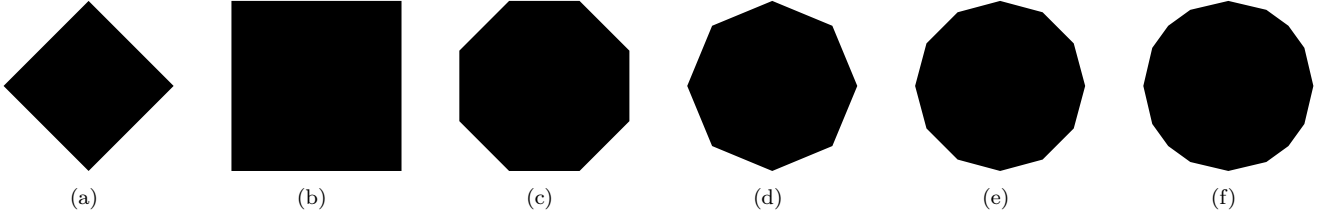


Figure 1. Balls with (a) $\alpha, \beta, \gamma = 1, 1, 1$ [city-block], (b) $\alpha, \beta, \gamma = 1, 1, 0$ [chessboard], (c) $\alpha, \beta, \gamma = 1, 1, \frac{\sqrt{2}}{1+\sqrt{2}}$ [n.s.-distance], (d) $\alpha, \beta, \gamma = 1, \sqrt{2}, 0$ [weighted distance with two weights], (e) $\alpha, \beta, \gamma = 1, 3 - \sqrt{3}, \frac{1}{3}(3 - \sqrt{3})$ [weighted n.s.-distance], and (f) $\alpha, \beta, \omega = 1, \sqrt{2}, \sqrt{5}$ [weighted distance with three weights].

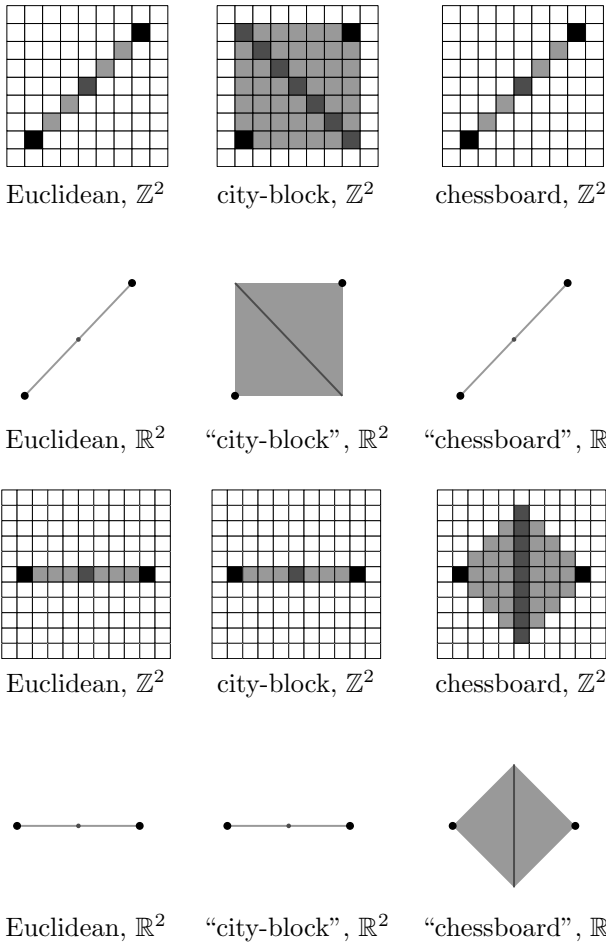


Figure 2. The sets S_p (defined in the text) are shown in grey. The sets $Q_{p,s}$ for $s = d(0, \mathbf{p})/2$ are shown in dark grey. Top rows: $\mathbf{p} = (6, 6)$. Bottom rows: $\mathbf{p} = (8, 0)$. The points $(0, 0)$ and \mathbf{p} are colored black.

a point) in the continuous case. See Figure 2. For the distance functions defined on \mathbb{R}^2 , the following error

function is considered:

$$E = \max_{s : 0 \leq s \leq r} |Q_{\mathbf{p},s}|, \quad (4)$$

$$\mathbf{p} : d(\mathbf{0}, \mathbf{p}) = r$$

where $|Q_{\mathbf{p},s}|$ is the (Euclidean) length of the line $Q_{\mathbf{p},s}$. The distance functions (defined for \mathbb{R}^2) we consider here are $d_{\alpha,\beta}(\cdot, \cdot, \gamma)$ and $d_{\alpha,\beta,\omega}(\cdot, \cdot)$ for fixed values of α, β, ω , and γ . The value of E is shown in Table 1 for some distance functions. The error function verifies the result known from [11], i.e. that by combining n.s. and weights, a smaller difference with the Euclidean distance can be obtained than if only n.s. or weights are used. Moreover, from Table 1 it is clear that using weighted distance with three weights gives an even smaller difference, but that the gain is limited.

5 The Constrained Distance Transform

We use the algorithm from [11] with a small modification to apply also to weighted distance with three weights, [8], to compute the CDT resulting in Algorithm 5.1.

Two points \mathbf{p} and \mathbf{q} are *connected* if there is a digital straight line between \mathbf{p} and \mathbf{q} consisting of only object pixels, see [8]. The points that a point \mathbf{p} propagate distance values to must be connected with \mathbf{p} to avoid that the minimal cost-path cross thin obstacles.

To find the minimal cost-path between a pixel \mathbf{p} and a source pixel $\mathbf{0}$, we start by calculating the CDT for the source $\mathbf{0}$ using Algorithm 5.1. The minimal cost-path is then found by walking backwards from \mathbf{p} in the direction of the steepest gradient on the CDT, taking the weights and neighbourhood sequence into account. See Algorithm 5.2. If more than one source pixel is used, the algorithm finds a minimal cost path to the closest source pixel.

Table 1. Values of E for some distance functions.

$\alpha, \beta, \gamma, \omega$	Corresponding distance in \mathbb{Z}^2	E
1, 1, 0, -	chessboard distance	r
1, 1, 1, -	city-block distance	$\frac{\sqrt{2}}{2}r \approx 0.71r$
1, 1, $\frac{\sqrt{2}}{1+\sqrt{2}}$, -	n.s.-distance	$(\sqrt{2} - 1)r \approx 0.41r$
1, $\sqrt{2}$, 0, -	weighted distance	$\frac{\sqrt{2-\sqrt{2}}}{2}r \approx 0.38r$
1, $3 - \sqrt{3}$, $\frac{3-\sqrt{3}}{3}$, -	weighted n.s.-distance	$\sin\left(\frac{\pi}{12}\right)r \approx 0.26r$
1, $\sqrt{2}$, -, $\sqrt{5}$	weighted distance with three weights	$\sin\left(\frac{\arctan(1/2)}{2}\right)r \approx 0.23r$

Algorithm 5.1 Computing CDT for weighted n.s. distances

Input: *Either*

◊ B where $b(i) \in \{1, 2\} \forall i, \alpha, \beta$ or

◊ $B = (3), \alpha, \beta, \omega$,

an object $S \subset \mathbb{Z}^2$, and a source pixels $\mathbf{0}$.

Output: *The distance transform, DT_{cost} , and the number of steps in the corresponding minimal cost paths, DT_{length} .*

Notation: $w_{\mathbf{p}, \mathbf{q}}$ is α if \mathbf{p}, \mathbf{q} are 1-neighbours, β if \mathbf{p}, \mathbf{q} are strict 2-neighbours, and ω if \mathbf{p}, \mathbf{q} are strict 3-neighbours.

Auxiliary data structures: *A list \mathcal{L} of active pixels.*

Initialisation: *Set $DT_{cost}(\mathbf{0}) \leftarrow 0$ and $DT_{cost}(\mathbf{p}) \leftarrow \infty$ for all other pixels $\mathbf{p} \in S$. Set $DT_{length} \leftarrow DT_{cost}$. Insert the source pixel $\mathbf{0}$ in \mathcal{L} .*

while \mathcal{L} is not empty

Find \mathbf{p} in \mathcal{L} with lowest $DT_{cost}(\mathbf{p})$

Remove \mathbf{p} from \mathcal{L}

forall $\mathbf{q} \in S$ that are $b(DT_{length}(\mathbf{p}) + 1)$ -neighbours and connected with \mathbf{p} :

if $DT_{cost}(\mathbf{p}) + w_{\mathbf{p}, \mathbf{q}} < DT_{cost}(\mathbf{q})$

$DT_{cost}(\mathbf{q}) \leftarrow DT_{cost}(\mathbf{p}) + w_{\mathbf{p}, \mathbf{q}}$

$DT_{length}(\mathbf{q}) \leftarrow DT_{length}(\mathbf{p}) + 1$

Insert \mathbf{q} in \mathcal{L}

endif

endfor

endwhile

In Figure 3, the effect of the choice of distance function for computing the CDT is illustrated. Top, the obstacles together with the source pixel $\mathbf{0}$ and the goal pixel \mathbf{p} (left) and the shortest Euclidean path between $\mathbf{0}$ and \mathbf{p} (right) are shown. In the remaining subfigures, all possible minimal cost-paths between $\mathbf{0}$ and \mathbf{p} for some distance functions are shown. Approximations of the optimal values in Table 1 is used. The n.s.-distance defined by $(\alpha, \beta) = (1, 1)$ and $B = (1, 2)$ with (asymptotically) $\gamma = 0.5$ is used to approximate the optimal value $\gamma \approx 0.59$. The approximations $(\alpha, \beta) = (3, 4)$ of the optimal weights for the weighted distance is used

in, e.g. [1]. The approximations used in Figure 3 for weighted n.s.-distance and weighted distance with three weights are from [11] and [1], respectively.

Algorithm 5.2 Finding a minimal cost path between two pixels \mathbf{p} and $\mathbf{0}$.

Input: $\alpha, \beta, (\omega), B, DT_{cost}, DT_{length}$, and an object $S \subset \mathbb{Z}^2$ as described in Algorithm 5.1. A start point $\mathbf{p} \in S$.

Output: *A minimal cost path \mathcal{P} from \mathbf{p} to $\mathbf{0}$.*

Notation: $S(\mathbf{p})$ is the set of points connected to \mathbf{p} such that \mathbf{q} and \mathbf{p} are $b(DT_{length}(\mathbf{q}) + 1)$ -neighbours $\forall \mathbf{q} \in S(\mathbf{p})$.

Insert \mathbf{p} in \mathcal{P} and set $\mathbf{p}' \leftarrow \mathbf{p}$

while $DT_{cost}(\mathbf{p}') \neq 0$

$\mathbf{q} \leftarrow \min_{\mathbf{q}' \in S(\mathbf{p}')} (DT_{cost}(\mathbf{q}') + w_{\mathbf{p}', \mathbf{q}'})$

Insert \mathbf{q} in \mathcal{P} and set $\mathbf{p}' \leftarrow \mathbf{q}$

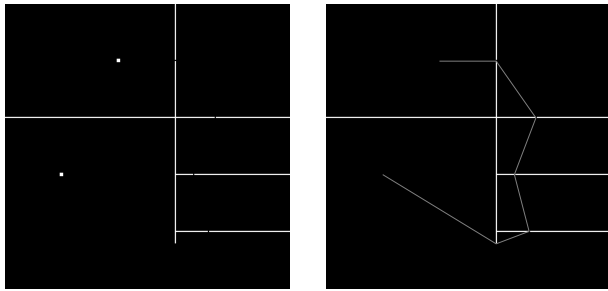
endwhile

The obstacles are chosen to visualize directions where the distance function is non favorable, i.e. where the deviation from the shortest Euclidean path is the largest. For both weighted n.s.-distance with two weights and weighted distance with three weights, the deviation is small from the shortest Euclidean path, which verifies the result shown in Table 1.

6 Discussion

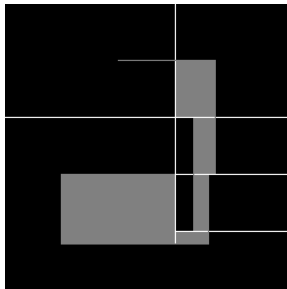
In this paper, we have given some aspects on the problem of finding *one* minimal cost-path \mathcal{P} between two points \mathbf{p} and \mathbf{q} . If there are several minimal cost-paths between \mathbf{p} and \mathbf{q} , the minimal cost-path \mathcal{P} may have large deviation from a straight (Euclidean) line between \mathbf{p} and \mathbf{q} . The performance of the path-based distance functions have been evaluated using a new error function, introduced for distance functions defined for \mathbb{R}^2 in this paper. The error function links the number of possible minimal cost-paths with the shape of the balls.

We have shown that the number of minimal cost-paths is almost as small for the optimal choice of weighted n.s.-distance with two weights (3×3 neigh-



Constrained image

Euclidean distance

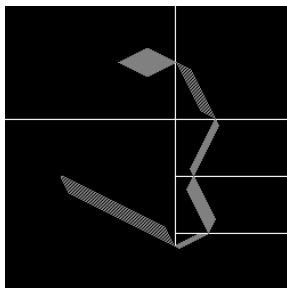


cityblock

$B = (1), (\alpha, \beta) = (1, 1)$

chessboard

$B = (2), (\alpha, \beta) = (1, 1)$

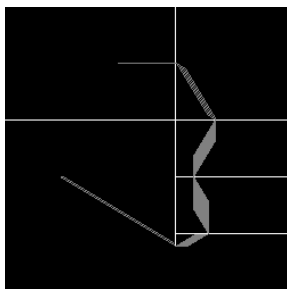


n.s.-distance

$B = (1, 2), (\alpha, \beta) = (1, 1)$

weighted distance

$B = (2), (\alpha, \beta) = (3, 4)$



weighted n.s.-distance

$B = (1, 2, 1, 2, 2),$
 $(\alpha, \beta) = (4, 5)$

weighted distance

with three weights
 $(\alpha, \beta, \omega) = (5, 7, 11)$

Figure 3. The minimal cost-paths for some distance functions (in \mathbb{Z}^2).

bourhood) as for weighted distance with three weights (5×5 neighbourhood). Using three weights gives a higher computational complexity since we need to check all strict 3-neighbours for connectedness when

propagating distances, [8]. This is not needed when two weights are used, since any two 2-neighbours are connected. Weighted n.s.-distances with two weights, on the other hand, have slightly higher memory requirements since both the length and the cost of the paths must be stored, resulting in two images (DT_{cost} and DT_{length}). For weighted distances with three weights, only one image is needed since B is constant so DT_{length} is not needed in Algorithm 5.1 and 5.2.

Acknowledgements

Stina Svensson is financially supported by Swedish Research Council (project 621-2005-5540).

References

- [1] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344–371, 1986.
- [2] D. Coeurjolly, S. Miguet, and L. Tougne. 2D and 3D visibility in discrete geometry: an application to discrete geodesic paths. *Pattern Recognition Letters*, 25(5):561–570, 2004.
- [3] A. X. Falcão, J. K. Udupa, and F. K. Miyazawa. An ultra-fast user-steered image segmentation paradigm: Live wire on the fly. *IEEE Transactions on Medical Imaging*, 19:55–62, 2000.
- [4] M. W. Jones, J. A. Baerentzen, and M. Sramek. 3D distance fields: a survey of techniques and applications. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):581–599, 2006.
- [5] G. Levi and U. Montanari. A grey-weighted skeleton. *Information and Control*, 17:62–91, 1970.
- [6] K. Norell, J. Lindblad, and S. Svensson. Gray weighted polar distance transform for outlining circular and approximately circular objects. Accepted for publication in *Proceedings of ICIAP 2007* (IEEE Computer Society), 2007.
- [7] F. Ortiz, S. Puente, and F. Torres. Mathematical morphology and binary geodesy for robot navigation planning. In S. Singh et al., editor, *Pattern Recognition and Data Mining, ICAPR 2005*, volume 3686 of *LNCS*, pages 118–126. Springer, 2005.
- [8] J. Piper and E. Granum. Computing distance transformations in convex and non-convex domains. *Pattern Recognition*, 20(6):599–615, 1987.
- [9] A. Rosenfeld and J. L. Pfaltz. Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494, 1966.
- [10] A. Rosenfeld and J. L. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1:33–61, 1968.
- [11] R. Strand. Weighted distances based on neighbourhood sequences. Accepted for publication in *Pattern Recognition Letters*, 2007.
- [12] B. J. H. Verwer. Local distances for distance transformations in two and three dimensions. *Pattern Recognition Letters*, 12(11):671–682, Nov. 1991.
- [13] B. J. H. Verwer, P. W. Verbeek, and S. T. Dekker. An efficient uniform cost algorithm applied to distance transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(4):425–429, 1989.