Weighted Distance Transforms for Images Using Elongated Voxel Grids

Ida-Maria Sintorn and Gunilla Borgefors

Centre for Image Analysis, Swedish University of Agricultural Sciences
Lägerhyddvägen 17, SE-752 37 Uppsala, Sweden
{ida,gunilla}@cb.uu.se

Abstract. In this paper we investigate weighted distance transforms in 3D images using elongated voxel grids. We use a local neighbourhood of size $3 \times 3 \times 3$ and assume a voxel grid with equal resolution along two axes and lower along the third. The weights (local distances) in the local neighbourhood are optimized by minimizing the maximum error over linear trajectories, which is a completely digital approach. General solutions are presented, as well as numerical solutions for the cases when the voxels are 1.5 and 2.58 times longer in one direction. Integer solutions for both real and integer scale factors are presented. As an application example, the medial axis of an object is computed in an image with elongated voxels and compared to the medial axis computed on the same image interpolated to equal resolution along all axes.

1 Introduction

Three-dimensional imaging systems often generate images with unequal resolution along the different axes. The case investigated here is when the resulting images consist of elongated voxels, where two sides are equal and the third is longer. This is generally true for tomographic, and many microscopic, images. To be able to apply standard image operations, the images are usually interpolated in the longer direction to give cubic voxels. This interpolation step results in a much larger data set without adding any new information compared to the original image. It also vastly increases the computational load for any operation applied.

Distance transforms (DTs), are used in many various image operations. Therefore, a DT that could be applied to the original image with elongated voxels would be a useful tool. A distance transformation is applied to a binary image, where it computes the distance from each voxel in the foreground (background) to the nearest voxel in the background (foreground). The image is converted to a grey-level image, as each voxel in the foreground (background) is given the value of the distance. As in 2D, the DT is computed by propagating local distances over the image using two raster scans [7]. When computing a weighted DT (WDT) the local distances used (in our case those in a $3 \times 3 \times 3$ mask) are given different weights, so that the resulting DT becomes as rotation independent as possible.

A. Braquelaire, J.-O. Lachaud, and A. Vialard (Eds.): DGCI 2002, LNCS 2301, pp. 244–254, 2002.

The Euclidean distance is rotation independent up to digitization effects. Therefore, the weights in the WDT are chosen to minimize the difference between the WDT and the Euclidean distance. An Euclidean distance transform for images with elongated voxels (or pixels) is possible to construct, but we have not seen it in the literature. In any case, an Euclidean distance transform is inconvenient for many applications and it is also more computationally demanding than the WDT. Integer approximations of the weights in the WDT are often calculated, since integers are usually preferred in image processing.

WDTs in square and cubic grids have been thoroughly investigated [2,3]. For rectangular grids, WDTs have been optimized using two different optimization criteria. Coquin and Bolon [4], use the difference between the Euclidean distance and the weighted distance over circular trajectories as their optimization criterion. Bolon, Vila, and Auzepy [1], and Sintorn and Borgefors [8], optimize over linear trajectories. In three dimensions, WDTs in parallelepipedic grids have so far only been optimized using circular trajectories, [5], but without any discussion of constraints on the local distances, or of the different cases that occur. (Not all combinations of local distances give the expected results.) In this paper, we optimize over linear trajectories, which is a completely digital approach. In Section 2, we describe our optimization method and in Section 3, we give resulting WDTs and show an application example.

2 Method

The working space here is a binary 3D image digitized in an elongated voxel grid. Each voxel has height and width equal to 1, and length $\Lambda \geq 1$. This is the most common case in tomographic and microscopic images. Fig. 1 shows the local distances in the first octant of a $3 \times 3 \times 3$ neighbourhood. The optimal local distances a, b, c, d, and e need to be determined for each Λ. Our optimization criterion is to minimize the maximum error between the $3 \times 3 \times 3$ WDT and the digital Euclidean distance in a cubic image of size $\Lambda M \times \Lambda M \times M$, see Fig. 2.

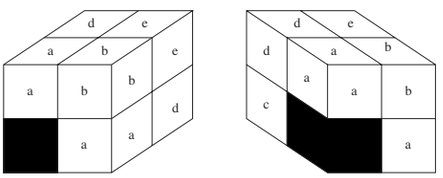

Fig. 1. Local distances for $3 \times 3 \times 3$ WDTs in rectangular grids. (Two views of the same neighbourhood.)

All sets of local distances do not give useful WDTs. The shortest path between two voxels is the set of local distances used to connect the voxels that gives the minimal total cost (sum of weights). Some constraints need to

be applied to a, b, c, d, and e to prohibit the shortest path between two voxels to zig-zag over the image, which would result in extremely complex distance expressions and serious violations of the triangular inequality for the resulting distance measure.

Definition 1: Assume that two voxels can be connected by only one type and one direction of local steps. If these steps constitute a minimal path defining the distance, the WDT is **semi-regular**. If it is the *only* minimal path the WDT is **regular**.

If Λ is assumed to be ≥ 1, the following constraints need to be fulfilled for the WDT to be semi-regular:

$$a \leq b, \quad a \leq c, \quad b \leq d, \quad c \leq d, \quad d \leq e, \quad 2e \leq b+2d, \quad b \leq 2a, \quad d \leq a+c. \tag{1}$$

These inequalities are not sufficient to determine unique expressions for the distance values. The path from the origin to the points (1,2,1) and (1,1,2) can be formed in two ways fulfilling the regularity criteria stated above. The point (1,2,1) can be reached either by steps $a+b$ or $d+b$, and the point (1,1,2) can be reached by steps $e+c$ or $d+d$. These ambiguities give rise to four cases, which are listed below. The only case further investigated here is denoted $Case$ 1.

$$\begin{array}{lll} Case\ 1: e+a \leq d+b & \text{and} & e+c \leq 2d, \\ Case\ 2: e+a \leq d+b & \text{and} & 2d \leq e+c, \\ Case\ 3: d+b \leq e+a & \text{and} & e+c \leq 2d, \\ Case\ 4: d+b \leq e+a & \text{and} & 2d \leq e+c. \end{array} \tag{2}$$

Optimization is performed in one half of the first octant, with one object voxel in (0,0,0). Due to symmetry, the rest of the volume need not be considered. The seven other octants are symmetric to the first one, and the two halves of the first octant are symmetric since the case when $x = \Lambda M$ is symmetric to the case when $y = \Lambda M$. The expression for the computed distance is different in different parts of the volume. The three distance expressions become:

$$\begin{array}{lll} DT(x,y,z) = ax + (b-a)y + (e-b)z & \text{for} & 0 \leq z \leq y \leq x, \\ DT(x,y,z) = ax + (d-a)z + (e-d)y & \text{for} & 0 \leq y \leq z \leq x, \\ DT(x,y,z) = cz + (d-c)x + (e-d)y & \text{for} & 0 \leq y \leq x \leq z. \end{array} \tag{3}$$

We can assume that the maximal difference to the EDT occurs on the border of the image, i.e., for $x = \Lambda M$ or $z = M$. This results in five difference equations. Fig. 2 shows the areas that need to explored and also were the five difference equations occur.

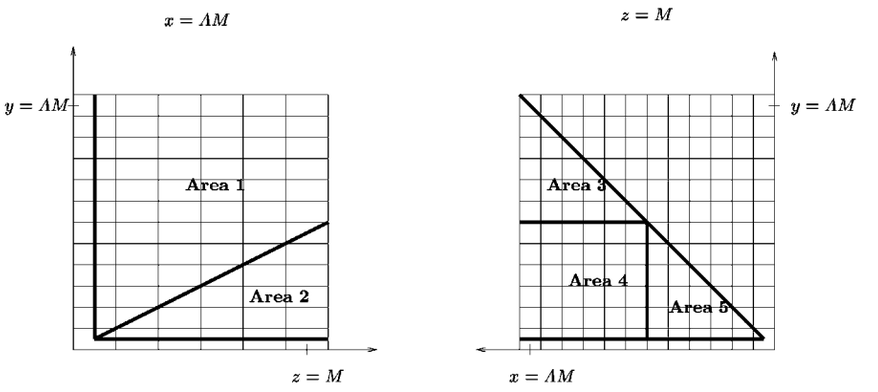

Fig. 2. Geometry of the WDT in the $3 \times 3 \times 3$ neighbourhood case. Two orthogonal cuts through an $\Lambda M \times \Lambda M \times M$ image are shown.

$$
\begin{aligned}
&\mathrm{Diff}_1(\Lambda M, y, z) = a\Lambda + (b-a)y + (e-b)z - \sqrt{\Lambda^2 M^2 + y^2 + \Lambda^2 z^2} \\
&\quad \text{for} \quad 0 \le z \le M, z \le y \le \Lambda M = \mathit{Area}\ 1, \\
&\mathrm{Diff}_2(\Lambda M, y, z) = a\Lambda M + (d-a)z + (e-d)y - \sqrt{\Lambda^2 M^2 + y^2 + \Lambda^2 z^2} \\
&\quad \text{for} \quad 0 \le y \le M, y \le z \le M = \mathit{Area}\ 2, \\
&\mathrm{Diff}_3(x, y, M) = ax + (b-a)y + (e-b)M - \sqrt{x^2 + y^2 + \Lambda^2 M^2} \\
&\quad \text{for} \quad M \le x \le \Lambda M, M \le y \le x = \mathit{Area}\ 3, \\
&\mathrm{Diff}_4(x, y, M) = ax + (d-a)M + (e-d)y - \sqrt{x^2 + y^2 + \Lambda^2 M^2} \\
&\quad \text{for} \quad M \le x \le \Lambda M, 0 \le y \le M = \mathit{Area}\ 4, \\
&\mathrm{Diff}_5(x, y, M) = cM + (d-c)x + (e-d)y - \sqrt{x^2 + y^2 + \Lambda^2 M^2} \\
&\quad \text{for} \quad 0 \le x \le M, 0 \le y \le x = \mathit{Area}\ 5.
\end{aligned}
\tag{4}
$$

To optimize the WDT, we must find the a, b, c, d, and e that minimize the maximum of these expressions. For each of them the maximum can occur at the end points of the intervals, or where the derivative is equal to zero. This results in 39 error equations, of which 23 are different. These are listed below. The top index shows in what area the error equation appears, and the lower index is a consecutive numbering of all error equations. All E_j^i have been divided by M.

Area 1 :

$$
\begin{aligned}
&E_1^1 = \Lambda(a-1) \quad \text{for} \quad (\Lambda M, 0, 0), \\
&E_2^1 = \Lambda(a - \sqrt{1-(b-a)^2} \quad \text{for} \quad (\Lambda M, y_{\max}, 0), \\
&E_3^1 = \Lambda(b - \sqrt{2}) \quad \text{for} \quad (\Lambda M, \Lambda M, 0), \\
&E_4^1 = \Lambda(a - \sqrt{1-(e-a)^2/(1+\Lambda^2)} \quad \text{for} \quad (\Lambda M, z_{\max}, z_{\max}), \\
&E_5^1 = \Lambda a - \sqrt{\Lambda^2 - \Lambda^2(b-a)^2 - (e-b)^2} \quad \text{for} \quad (\Lambda M, y_{\max}, z_{\max}), \\
&E_6^1 = \Lambda b - \sqrt{2}\sqrt{\Lambda^2 - (e-b)^2} \quad \text{for} \quad (\Lambda M, \Lambda M, z_{\max}), \\
&E_7^1 = \Lambda a + e - a - \sqrt{2\Lambda^2 + 1} \quad \text{for} \quad (\Lambda M, M, M), \\
&E_8^1 = \Lambda a + e - b - \Lambda\sqrt{2}\sqrt{1-(b-a)^2} \quad \text{for} \quad (\Lambda M, y_{\max}, M), \\
&E_9^1 = \Lambda(b - \sqrt{3}) + e - b \quad \text{for} \quad (\Lambda M, \Lambda M, M).
\end{aligned}
\tag{5}
$$

Area 2 :
$$
\begin{aligned}
&E^2_{10} = E^1_1,\\
&E^2_{11} = \Lambda a - \sqrt{\Lambda^2 - (d-a)^2} \quad \text{for} \quad (\Lambda M, 0, z_{\max}),\\
&E^2_{12} = \Lambda(a - \sqrt{2} + d - a \quad \text{for} \quad (\Lambda M, 0, M),\\
&E^2_{13} = E^1_4,\\
&E^2_{14} = \Lambda a - \sqrt{\Lambda^2 - \Lambda^2(e-d)^2 - (d-a)^2} \quad \text{for} \quad (\Lambda M, y_{\max}, z_{\max}),\\
&E^2_{15} = \Lambda a + d - a - \Lambda\sqrt{2}\sqrt{1-(e-d)^2} \quad \text{for} \quad (\Lambda M, y_{\max}, M),\\
&E^2_{16} = E^1_7.
\end{aligned}
\tag{6}
$$

Area 3 :
$$
\begin{aligned}
&E^3_{17} = e - \sqrt{\Lambda^2 + 2} \quad \text{for} \quad (M, M, M),\\
&E^3_{18} = \emptyset,\\
&E^3_{19} = E^1_7 = E^2_{16},\\
&E^3_{20} = \emptyset,\\
&E^3_{21} = \emptyset,\\
&E^3_{22} = E^1_8,\\
&E^3_{23} = E^1_9.
\end{aligned}
\tag{7}
$$

Area 4 :
$$
\begin{aligned}
&E^4_{24} = d - \sqrt{\Lambda^2 + 1} \quad \text{for} \quad (M, 0, M),\\
&E^4_{25} = \emptyset,\\
&E^4_{26} = E^2_{12},\\
&E^4_{27} = d - \sqrt{(1+\Lambda^2)(1-(e-d)^2)}, \quad \text{for} \quad (M, y_{\max}, M),\\
&E^4_{28} = \emptyset,\\
&E^4_{29} = E^2_{15},\\
&E^4_{30} = E^3_{17},\\
&E^4_{31} = \emptyset,\\
&E^4_{32} = E^1_7 = E^2_{16} = E^3_{19}.
\end{aligned}
\tag{8}
$$

Area 5 :
$$
\begin{aligned}
&E^5_{33} = c - \Lambda \quad \text{for} \quad (0, 0, M),\\
&E^5_{34} = c - \Lambda\sqrt{1-(d-c)^2} \quad \text{for} \quad (x_{\max}, 0, M),\\
&E^5_{35} = E^4_{24},\\
&E^5_{36} = c - \Lambda/\sqrt{2}\sqrt{2-(e-c)^2} \quad \text{for} \quad (y_{\max}, y_{\max}, M),\\
&E^5_{37} = c\Lambda\sqrt{1-(d-c)^2-(e-d)^2} \quad \text{for} \quad (x_{\max}, y_{\max}, M),\\
&E^5_{38} = E^4_{27},\\
&E^5_{39} = E^3_{17} = E^4_{30}.
\end{aligned}
\tag{9}
$$

Numerical experimentation shows that the minimum of these expressions occurs when

$$
-E^1_1 = -E^1_3 = E^1_5 = -E^3_{17} = -E^4_{24} = -E^5_{33}. \tag{10}
$$

Solving these equations give the following expressions for the optimal a, b, c, d, and e, and the maximal difference from the Euclidean distance, denoted maxdiff.

$$
\begin{aligned}
&a(\Lambda) = \frac{3\Lambda^2 + \Lambda(\sqrt{2} - 2 - \sqrt{\Lambda^2 + 2}) + \sqrt{\Lambda^2 + 2} + 1 - \sqrt{2}}{5\Lambda^2 - 2\Lambda + 1} + \\
&\frac{\sqrt{\Lambda^2(4\sqrt{\Lambda^2 + 2}(2\sqrt{2} - 1 + \Lambda) + 5\Lambda^2(2\sqrt{2} - 3) + 2\Lambda(3 - 4\sqrt{2}) + 6\sqrt{2} - 19)}}{5\Lambda^2 - 2\Lambda + 1}, \\
&b(\Lambda) = a(\Lambda) - 1 + \sqrt{2}, \\
&c(\Lambda) = \Lambda a(\Lambda), \\
&d(\Lambda) = \Lambda a(\Lambda) - \Lambda + \sqrt{1 + \Lambda^2}, \\
&e(\Lambda) = \Lambda a(\Lambda) - \Lambda + \sqrt{\Lambda^2 + 2}, \\
&\\
&\text{maxdiff}(\Lambda) = \mid \Lambda - \Lambda a(\Lambda) \mid .
\end{aligned}
\tag{11}
$$

These solutions are valid for all $\Lambda \geq 1$. With $\Lambda = 1$ this becomes exactly the solution for the cubic grid found in [3].

The solution when $a \equiv 1$ is needed when integer approximations with integer scale factors are calculated. The system to be solved then becomes

$$
-E_3^{*1} = E_5^{*1} = -E_{17}^{*3} = -E_{24}^{*4} = -E_{33}^{*5}, \tag{12}
$$

which gives:

$$
\begin{aligned}
&a^*(\Lambda) = 1 \\
&b^*(\Lambda) = \frac{2\sqrt{2}\Lambda^2 - (\sqrt{\Lambda^2 + 2} + \sqrt{2})\Lambda + \sqrt{\Lambda^2 + 2}}{3\Lambda^2 - 2\Lambda + 1} + \\
&\frac{\sqrt{\Lambda^2(6\sqrt{2} - 9)\Lambda^2 + 2\sqrt{2}\sqrt{\Lambda^2 + 2} - 4\sqrt{2} + 2)\Lambda - 7 + 2\sqrt{2}\sqrt{\Lambda^2 + 2} + 2\sqrt{2})}}{3\Lambda^2 - 2\Lambda + 1}, \\
&c^*(\Lambda) = \Lambda(b^* + 1 - \sqrt{2}), \\
&d^*(\Lambda) = \Lambda(b - \sqrt{2}) + \sqrt{\Lambda^2 + 1}, \\
&e^*(\Lambda) = \Lambda(b - \sqrt{2}) + \sqrt{\Lambda^2 + 2}, \\
&\\
&\text{maxdiff}(\Lambda) = \mid \Lambda(\sqrt{2} - b) \mid .
\end{aligned}
\tag{13}
$$

Note that we do not get $c^*(1) = a^*(1)$, i.e., we do not get the same solution as for the cubic grid for $\Lambda = 1$. This is correct. We have only constrained a^*, not c^*. With $a^* = 1$, the other local distances, including c^*, decrease to compensate for the forced increase in a^*.

Integer approximations are found by multiplying the optimal local distances by a scale factor and rounding to the nearest integer. If only integers are considered as scale factors the solution with $a \equiv 1,\ b^*,\ c^*,\ d^*,\ e^*$, is used, since a then will serve as the scale factor. Besides integer scale factors for the $a \equiv 1$ solutions, real scale factors ranging from 1 to 20, with three decimals, are investigated here, as integer scale factors give poor results for $\Lambda > 1$. In this case the solution $a(\Lambda),\ b(\Lambda),\ c(\Lambda),\ d(\Lambda)$, and $e(\Lambda)$ should be used.

In practice, for each integer approximation of a an integer neighbourhood of b is checked, and for each integer approximation of b an integer neighbourhood of c, etc, is searched. For each set of integer local distances the maximal difference is computed from the error equations $(5) - (9)$. Each solution is, of course, also examined to make sure that it fulfills the regularity criteria (1) and (2), $Case$ 1.

3 Results

In this section Tables of solutions and an illustrating example are presented. In Tables 1 and 2 the optimal, real, and integer local distances for $\Lambda = 1.5$, and $\Lambda = 2.58$ are shown, respectively. The sizes of Λ were chosen to represent a small and a fairly large Λ. $\Lambda = 2.58$ is also the Λ occurring in the image used in the application example. In the Tables, the solutions for a_{opt}, and $a \equiv 1$ are presented, as well as the best integer solutions for integer and real scale factors up to 20. Only solutions better than those for lower scale factors are listed. As can be seen by comparing the Tables, the error (maxdiff) grows with increasing Λ, as can be expected. It is interesting to compare our optimal values to those in [5], which optimizes the maximal error over circular trajectories. Their optimal local distances for $\Lambda = 1.5$ become: $a = 0.9237,\ b = 1.3063,\ c = 1.3855,\ d = 1.6652,\ e = 1.9042$, and for $\Lambda = 2.58$: $a = 0.8953,\ b = 1.2662,\ c = 2.3099,\ d = 2.4774,\ e = 2.6342$. As can be seen in the Tables, the two methods generate very similar local distances. A comparison of the errors for the two methods is irrelevant, since they use different optimization criteria.

As mentioned above, the error grows rapidly with increasing Λ. This means that the calculated weighted distances differ quite a lot compared to the Euclidean distances. Hence, this WDT is not a good choice when true distances are desired. The WDT also becomes more rotation dependent than desired. In many applications, however, only relative distances are needed and rotation is not an issue. The use of parallelepipedic WDTs can then save much time and memory. As an example, the medial axis (MA), also known as the set of centres of maximal balls, of an object is calculated on an image with different resolution along the z-axis as well as on the same image interpolated to cubic voxels. The MA in turn, can, e.g. be used for image description, analysis, and compression. The binary test image comes from a magnetic resonance angiography image, where the arteries have been segmented. The size of the original image is $280 \times 430 \times 96$ voxels (11.6 Mbyte) and each voxel is of size $1 \times 1 \times 2.58\ mm^3$. Interpolated to equal resolution along all three axes the image size becomes $280 \times 430 \times 250$ voxels (30Mbyte). The number of voxels in the original image is 416,423 compared to 1,084,929 for the interpolated image. Fig. 3 shows projections, middle slices, and middle slices of the MA of the interpolated and original images. All images are from the right $(z - y)$ side of the volume to show the difference in resolution in the z-direction. Hence, the big bend in the middle of the images is the aorta bending towards the person's back. Fig. 3a) and 3d) show simple 2D-projections of the volume, where the grey levels represent the depths of the object voxels.

Table 1. Integer approximations for $\Lambda = 1.5$.

scale factor	a	b	c	d	e	maxdiff
1.999	2	3	3	3	4	0.302
2.160	2	3	3	4	4	0.210
3.103	3	4	5	5	6	0.191
4.507	4	6	7	8	9	0.169
5.266	5	7	8	9	10	0.163
6.666	6	9	9	11	13	0.153
8.847	8	12	12	15	17	0.144
16.558	15	22	23	28	32	0.141
18.727	17	25	26	32	36	0.139
.	.	.	.	.	.	.
∞	a_{opt}	b_{opt}	c_{opt}	d_{opt}	e_{opt}	0.1372
opt	0.9085	1.3227	1.3628	1.6656	1.9243	0.1372
1	1	1	2	2	2	0.621
2	2	3	3	3	4	0.303
3	3	4	4	5	6	0.253
6	6	8	8	10	11	0.228
7	7	9	10	12	13	0.204
8	8	11	11	13	15	0.202
9	9	12	12	15	17	0.200
15	15	20	20	25	28	0.195
16	16	21	21	26	30	0.190
.	.	.	.	.	.	.
∞	1	$b*$	$c*$	$d*$	$e*$	0.1882
opt*	1	1.2887	1.3118	1.6145	1.8733	0.1882

For the interpolated image with equal resolution along all sides the common <3,4,5> WDT, in our notation 3-4-3-4-5, $maxdiff = 10.01\%$, was used, and for the original image with $\Lambda = 2.58$ the 6-9-16-17-18 WDT was chosen from Table 2. It was chosen because its error is rather close to the optimal error. To get a smaller error, the local distances have to be increased rather significantly, making the voxel values uncomfortably large.

The MA is computed from the WDT by local tests using a Look-up table, which is specific for each set of weights used. The Look-up tables store, for each voxel value of the WDT, the smallest values of the neighbours that prohibit the voxel to be part of the MA. The Look-up tables were calculated by the method presented by Remy and Thiel in [6]. To produce the Look-up tables, they first calculate the WDT from each point in the first 1/48th of the volume to the origin. The rest of the volume can be omitted from the calculations due to symmetry. Then they scan through this WDT and for each voxel and all directions of local steps they search for the smallest value that prohibit the voxel in focus to be part of the MA. This is done by comparing the value in the WDT at a position one mask step away from the voxel with the value of the voxel plus the weight of the mask step. Their method simultaneously tests whether the directions in

Table 2. Integer approximations for $\Lambda = 2.58$.

scale factor	a	b	c	d	e	maxdiff
1.999	2	3	4	5	5	0.584
2.301	2	3	6	6	6	0.338
4.601	4	6	11	12	12	0.337
6.882	6	9	16	17	18	0.331
17.105	15	22	39	42	45	0.330
.	.	.	.	.	.	.
∞	a_{opt}	b_{opt}	c_{opt}	d_{opt}	e_{opt}	0.3225
opt	0.8750	1.2892	2.2575	2.4445	2.6197	0.3225
1	1	1	2	2	2	1.069
2	2	3	4	5	5	0.582
3	3	4	7	7	8	0.546
4	4	5	9	10	11	0.481
6	6	8	13	14	15	0.446
8	8	10	18	19	20	0.442
15	15	19	33	35	38	440
19	19	24	41	45	48	0.435
.	.	.	.	.	.	.
∞	1	$b*$	$c*$		$d*$	$e*$ 0.4263
opt*	1	1.2490	2.1537	2.3408	2.5159	0.4263

the WDT mask are enough to produce the correct Look-up table, or if extra directions must be added to compute the table. The only changes that were needed to make the method applicable to a parallelepipedic grid was to modify the mask and to search 1/16th of the volume instead of 1/48th.

For the original image the MA consists of 110,067 voxels while the MA for the interpolated image consists of 207,013 voxels. The calculation of the MA of the original image required 1.12 CPUs compared to 2.83 CPUs for the interpolated image. Hence, it is almost exactly a factor Λ faster to calculate the MA in the original image instead of in the interpolated one. A further bonus is of course that no interpolation is needed.

4 Conclusions

Optimal weighted distance transforms for a $3\times3\times3$ neighbourhood in 3D images with elongated voxel grids have been investigated. The results presented are valid for all elongated voxel ratios as long as two sides are of equal length, which is often the case for tomographic and microscopic images. Numerical results for grids with voxels of size $1\times1\times\Lambda$, when Λ equals 1.5 and 2.58 have been presented. The new WDTs are useful in applications where relative distances are needed. As an example the MA was computed on a medical image with elongated voxels and compared to the MA computed on the same image interpolated to cubic voxels. To get a better approximation to the Euclidean distance transform a larger

Fig. 3. 2D-projection, middle slice, and middle slice of computed medial axis of segmented arteries from a magnetic resonance angiography image. All images are produced from the right $(z - y)$ side of the volume to show the difference in resolution in the z-direction. The lower row shows images from the original volume with voxel dimension $1 \times 1 \times 2.58$. The upper row shows images from the same volume but interpolated to voxel size $1 \times 1 \times 1$. The grey levels in c) and f) represent the voxel values in the MA. The resulting MA are quite similar in both cases (when interpolation is not considered).

neighbourhood could of course be considered. The asymmetrical neighbourhood $5 \times 5 \times 3$, used in [5], should then be a good way to compensate for the largest errors, without increasing the computation times as much as with a complete

$5 \times 5 \times 5$ neighbourhood. However, to calculate such local distances many more constraints would have to be considered to ensure that the WDTs are well-behaved (semi-regular) and that the equations used for optimization really are the ones valid in the *Case* investigated.

In summary, the errors become quite large when distance transforms are computed in images with elongated voxels, unless the elongation factor is small. In fact, a Λ of 1.57 is enough to double the optimal maxdiff compared to a cubic grid. The WDTs presented here are good choices for small Λ in all cases and for larger Λ in cases where rotation dependence and true distances are unimportant. In other cases, interpolation is the only feasible option.

References

1. P. Bolon, J. L. Vila, and T. Auzepy. Operateur local de distance en mailliage rectangulaire. In *Proc. 2eme Colloque de Geometrie Discrete en Imagerie: Fondements et Applications,* Grenoble, France, pages 45–56, Sept. 1992.
2. G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34:344–371, 1986.
3. G. Borgefors. On digital distance transforms in three dimensions. *Computer Vision and Image Understanding*, 64(3):368–376, Nov. 1996.
4. D. Coquin and P. Bolon. Discrete distance operator on rectangular grids. *Pattern Recognition Letters*, 16:911–923, 1995.
5. D. Coquin, Y. Chehadeh, and P. Bolon. 3D local distance operator on parallelepipedic grids. In *Proc. 4th Discrete Geometry for Computer Imagery,* Grenoble, France, pages 147–156, Sept. 1994.
6. E. Remy and E. Thiel. Computing 3d medial axis for chamfer distances. In Borgefors, Nyström, and S. di Baja, editors, *Discrete Geometry for Computer Imagery*, volume 1953 of *Lecture Notes in Computer Science*, pages 418–430. Springer-Verlag, Dec. 2000.
7. A. Rosenfeld and J. L. Pfaltz. Sequential operations in digital picture processing. *Journal of the Association for Computing Machinery*, 13(4):471–494, Oct. 1966.
8. I.-M. Sintorn and G. Borgefors. Weighted distance transforms in rectangular grids. In Ardizzone and Gesù, editors, *Proc. 11th International Conference on Image Analysis and Processing (ICIAP 2001),* Palermo, Italy, pages 322–326. IEEE Computer Society, sep 2001.