Euclidean skeletons of 3D data sets in linear time by the integer medial axis transform

Hesselink, Wim H.; Visser, Menno; Roerdink, Jos B.T.M.

Published in:
MATHEMATICAL MORPHOLOGY: 40 YEARS ON

Publication date:
2005

EUCLIDEAN SKELETONS OF 3D DATA SETS IN LINEAR TIME BY THE INTEGER MEDIAL AXIS TRANSFORM

Wim H. Hesselink, Menno Visser and Jos B.T.M. Roerdink[1]

[1] *Institute for Mathematics and Computing Science, University of Groningen, the Netherlands*
{wim,menno,roe}@cs.rug.nl

Abstract A general algorithm for computing Euclidean skeletons of 3D data sets in linear time is presented. These skeletons are defined in terms of a new concept, called the *integer medial axis* (*IMA*) transform. The algorithm is based upon the computation of 3D feature transforms, using a modification of an algorithm for Euclidean distance transforms. The skeletonization algorithm has a time complexity which is linear in the amount of voxels, and can be easily parallelized. The relation of the *IMA* skeleton to the usual definition in terms of centers of maximal disks is discussed.

Keywords: Feature transform, integer medial axis, 3-D Euclidean skeletonization.

1. Introduction

In computer vision, skeleton generation is often one of the first steps in image description and analysis. Intuitively, a skeleton consists of the center lines of an object, and therefore skeletons provide important structural information about image objects by a relatively small number of pixels.

There are four main approaches to skeletonization: 1) thinning, i.e. iterative removal of points from the boundary; 2) wave propagation from the boundary; 3) detection of crest points in the distance transformed image; 4) analytical methods. A large number of skeletonization algorithms exist, see e.g. [15], many of them based upon mathematical morphology [2, 10, 14, 17, 19, 20]. For a parallel 3D skeletonization algorithm based on thinning, see [9].

We note that in algorithms of type 3) one often restricts oneself to local maxima of the distance transform [18], but the resulting skeleton is far from the Euclidean one. The approach we present here is a variant of the third approach, using a definition of skeletons based on Blum's medial axis transform [3].

Often, one is satisfied with approximations to the Euclidean metric (e.g., using chamfer metrics). In 1980, Danielsson [6] gave two good approximating

C. Ronse et al. (eds.), Mathematical Morphology: 40 Years On, 259–268.

Euclidean distance transform algorithms, and applied them to obtain the centers of maximal (integer) disks (*CMD*), see below. He notes (p. 243) that application of skeletons has been hampered by the lack of true Euclidean distance maps. Especially in the 3D case where data size can be very large, many existing algorithms for computing 3D Euclidean skeletons are computationally too expensive [4]. Ge and Fitzpatrick [7] clearly identified the problem to determine the *CMD*: "The problems with existing methods lie in the discrepancies between continuous and discrete image maps". The paper [7] also mentions the goal of linking the centers of maximal disks into *connected* skeletons.

The main contribution of the present work is that we present a simple and easily parallelizable linear time algorithm which computes a skeleton defined in terms of a new concept, called the *integer medial axis* (*IMA*) transform. The algorithm works in arbitrary dimensions, and is based upon the general linear time Euclidean distance transform (EDT) algorithm of Hirata [8], which has been rediscovered several times, i.e., by ourselves, see Meijster *et al.* [13], and later by Maurer *et al.* [11, 12]. The skeletonization algorithm has two phases. First, a feature transform is computed, which uses essentially the same algorithm as for the distance transform, the difference being that not only distances are computed, but also the boundary points which realize the closest distance. The actual skeletonization is performed in a second pass through the data, where the integer medial axis is computed by assigning points to the skeleton depending on their feature transform.

Our method does not aim at a minimal skeleton useful for image compression with exact reconstruction, but at a computation of connected skeletons directly from the Euclidean feature transform, thus avoiding the costly and complicated phase of removing centers of not-quite-maximal disks by the techniques of [16]. We establish a number of mathematical properties of the *IMA* and point out some relations to Blum's real medial axis (*RMA*) and to the *CMD* skeleton. More work is needed to establish the topological characteristics of the *IMA* skeleton.

Often, simplification or pruning of the skeleton is used as a postprocessing step to remove unwanted points, which arise especially in noisy data [1]. In our approach, skeleton pruning can be handled in the algorithm itself, by a single adjustable parameter through which one can prune the skeleton during the second pass of the algorithm.

In order to derive our algorithm, we first modify the *EDT* algorithm of Meijster *et al.* to calculate 3D feature transforms, from which the *IMA* skeletons are derived. For all program parts, explicit and compact pseudocode is given.

2. Feature transform computation

We briefly describe extension of the Euclidean distance transform algorithm to the computation of feature transforms, closely adhering to the notation and approach given in [13]. The algorithm can deal with several types of distances (Manhattan, chessboard, or chamfer distances), but we will limit ourselves to the case of the Euclidean distance here, since we focus on Euclidean skeletons in this paper.

The length of a vector $\mathbf{r} \in \mathbb{R}^d$ is denoted by $||\mathbf{r}|| = \sqrt{\sum_i \mathbf{r}_i^2}$. We regard $\mathbb{Z}^d$ as a grid embeddded in $\mathbb{R}^d$. The elements of $\mathbb{Z}^d$ are called grid points.

Let B be the background, which is a given nonempty set of grid points. The Euclidean distance transform *dt* of B is the function that assigns to every grid point $\mathbf{r}$ the distance to the nearest background point, so $dt(\mathbf{r}, B) = \min\{||\mathbf{r} - \mathbf{y}|| \mid \mathbf{y} \in B\}$. The *feature transform FT* is defined as the set-valued function that assigns to $\mathbf{r}$ the set of closest boundary points. So we have $FT(\mathbf{r}, B) = \{\mathbf{y} \in B \mid ||\mathbf{r} - \mathbf{y}|| = dt(\mathbf{r}, B)\}$. The parameter B is omitted from *dt* and *FT* when it is clear from the context.

It is possible to compute *FT*, but it is computationally cheaper and sufficient for our purposes to compute, for every point $\mathbf{r}$, just a single feature transform point $ft(\mathbf{r})$. So, the function *ft* is incompletely specified by $ft(\mathbf{r}) \in FT(\mathbf{r})$. In fact, we compute $ft(\mathbf{r})$ as the first element of $FT(\mathbf{r})$ with respect to a lexical ordering.

The computation of *ft* proceeds in d phases. We specify the results of these phases as follows. For $0 < i \leq d$, let L_i be the i-dimensional subspace spanned by the first i standard basis vectors of $\mathbb{R}^d$. The i-th phase computes the i-dimensional feature transform ft_i which is characterized by $ft_i(\mathbf{r}) \in FT(\mathbf{r}, B \cap (\mathbf{r} + L_i))$. The result of the last phase is $ft = ft_d$. Since the components of $ft_i(\mathbf{r})$ orthogonal to L_i are always equal to the corresponding components of $\mathbf{r}$, we only compute and use the orthogonal projection of ft_i on L_i.

In Figures 1 and 2, we present the computation for the case $d = 3$ in a box of size (m, n, p). Since ft_i is a vector-valued function, the three components of $ft_i(\mathbf{r})$ are written $ft_i[\mathbf{r}].x, ft_i[\mathbf{r}].y$, and $ft_i[\mathbf{r}].z$.

The first phase is the computation of ft_1 given in Fig. 1. For every pair (y, z), it consists of two scans over the line $(0, y, z) + L_1$. The boundary B is represented here by a 3D boolean array b. In the first scan, $g[x]$ becomes the distance to the next boundary point along the line. The second scan collects ft_1.

The second and third phases are given in Fig. 2. In the body of the outer loop, the value of ft_i is computed from ft_{i-1} for a given scan line, again by two scans. The results of the forward scan are collected on stacks s and t, with common stack pointer q. The backward scan reaps ft_i as harvest. The auxiliary functions f and *Sep* are given by $f(i, u) = (i - u)^2 + g(u)$ and

$Sep(i,u) = (u^2 - i^2 + g(u) - g(i))$ **div** $(2(u-i))$, where the function g is the squared Euclidean distance transform of the previous phase. So, $g(i) = (x - ft_1[x,i,z].x)^2$ in phase 2, and $g(i) = (x - ft_2[x,y,i].x)^2 + (y - ft_2[x,y,i].y)^2$ in phase 3. Note that, in the body of the outer loop, we regard x and z as constants for phase 2, and x and y as constants for phase 3.

Since the algorithm is completely analogous to our algorithm for the Euclidean distance transform, we refer to paper [13] for further details.

```
forall y ∈ [0..n − 1], z ∈ [0..p − 1] do
  (∗ scan 1 ∗)
  if b[m − 1, y, z] then  g[m − 1] := 0
  else  g[m − 1] := ∞
  endif
  for x := m − 2 downto 0 do
    if b[x, y, z] then  g[x] := 0
    else  g[x] := 1 + g[x + 1]
    endif
  end for
  (∗ scan 2 ∗)
  ft1[0, y, z].x := g[0]
  for x := 1 to m − 1 do
    if x − ft1[x − 1, y, z].x ≤ g[x] then
      ft1[x, y, z].x := ft1[x − 1, y, z].x
    else
      ft1[x, y, z].x := x + g[x]
    endif
end forall
```

Figure 1. Program fragment for the first phase - one dimensional feature transform in 3D.

3. Skeletonization

The feature transform of a data set can be used to compute its skeleton. We first examine the definition of the medial axis [3], see also [5–7, 16]. Actually, we present three possible formalizations: *CMD*, *RMA*, and *IMA*. Since *RMA* is not restricted to grid points, whereas *CMD* and *IMA* are, the latter two are the main contenders.

The real medial axis and *CMD* skeleton. For the moment we assume that the boundary B is a closed subset of $\mathbb{R}^d$. For every point $x \in \mathbb{R}^d$, we can form the largest open disk $D(x,r) = \{y \in \mathbb{R}^d \mid ||x - y|| < r\}$ that is disjoint with B. This is called the *inscribed disk* of x. If an inscribed disk at point p is not contained in any other inscribed disk of B, we call it a *maximal disk* with center p. We define the *real medial axis RMA* to consist of the points $x \in \mathbb{R}^d \setminus B$ which are centers of maximal disks.

```
forall x ∈ [0..m − 1], z ∈ [0..p − 1] do
  q := 0; s[0] := 0; t[0] := 0
  for u := 1 to n − 1 do (∗ scan 1 ∗)
    while q ≥ 0 ∧ f(t[q], s[q]) > f(t[q], u) do
      q := q − 1
    if q < 0 then
      q := 0; s[0] := u
    else
      w := 1 + Sep(s[q], u)
      if w < n then
        q := q + 1; s[q] := u; t[q] := w
      endif
    endif
  end for
  for u := n − 1 downto 0 do  (∗ scan 2 ∗)
    ft2[x, u, z].x := ft1[x, s[q], z].x
    ft2[x, u, z].y := s[q]
    if u = t[q] then q := q − 1 endif
  end for
end forall
```

(a) Second phase

```
forall x ∈ [0..m − 1], y ∈ [0..n − 1] do
  q := 0; s[0] := 0; t[0] := 0
  for u := 1 to p − 1 do (∗ scan 1 ∗)
    while q ≥ 0 ∧ f(t[q], s[q]) > f(t[q], u) do
      q := q − 1
    if q < 0 then
      q := 0; s[0] := u
    else
      w := 1 + Sep(s[q], u)
      if w < p then
        q := q + 1; s[q] := u; t[q] := w
      endif
    endif
  end for
  for u := p − 1 downto 0 do  (∗ scan 2 ∗)
    ft3[x, y, u].x := ft2[x, y, s[q]].x
    ft3[x, y, u].y := ft2[x, y, s[q]].y
    ft3[x, y, u].z := s[q]
    if u = t[q] then q := q − 1 endif
  end for
end forall
```

(b) Third phase

Figure 2. Program fragments for the second and third phase.

For $x \in \mathbb{Z}^d$, the *inscribed integer disk* $M(x)$ is the intersection $D(x,r) \cap \mathbb{Z}^d$, where $D(x,r)$ is its inscribed disk. The set *CMD* (centers of maximal disks) consists of the points $x \in \mathbb{Z}^d$ for which $M(x)$ is not contained in any $M(y)$ with $y \neq x$, see also [7, 16]. As is presumably well known, it is not true that $\mathit{CMD} \subseteq \mathit{RMA} \cap \mathbb{Z}^d$.

EXAMPLE 1 *Let B consist of the four points $(0,0)$, $(3,0)$, $(0,3)$, and $(3,3)$. The intersection RMA$\cap\mathbb{Z}^d$ is empty, but CMD contains the points $(1,1)$, $(1,2)$, $(2,1)$, and $(2,2)$.*

Our aim is to define a skeleton that looks like the real medial axis of a smoothing of the boundary and tends to be connected when the complement of the boundary is connected, while still being computable in linear time.

Recall that $dt(x) = \min\{||x-y|| \mid y \in B\}$ and $\mathit{FT}(x) = \{y \in B \mid ||x-y|| = dt(x)\}$. Clearly, $dt(x)$ is the radius of the inscribed disk of x (for $x \in B$, we regard the empty set as an open disk with radius 0). The function $ft : \mathbb{R}^d \to B$ is incompletely specified by $ft(x) \in \mathit{FT}(x)$.

The next lemma may not be surprising, but it seems to be new.

LEMMA 2 *Assume B is a discrete (i.e., locally finite) subset of $\mathbb{R}^d$. Let $x \in \mathbb{R}^d$. Then $x \in$ RMA if and only if FT(x) has more than one element.*

This lemma is not true when B is not discrete. For example, in the case of an ellipse, the real medial axis is a segment of the long axis strictly inside of the ellipse; the two extremal points of the segment belong to *RMA* and yet have only one element in the feature transform set.

Henceforth, we assume that the boundary consists of grid points only, i.e. that $B \subseteq \mathbb{Z}^d$. It follows that B is discrete, so that Lemma 2 applies. The following result is almost trivial to verify, but it is quite useful.

LEMMA 3 *Let $x \in \mathbb{R}^d$ and let y, z be two different elements of FT(x). Then $||y - z|| \geq 1$. If moreover $x \in \mathbb{Z}^d$, then $||y - z|| > 1$.*

The integer medial axis. Since we assume the boundary now to consist of grid points only, *RMA* contains many points that would disappear when the boundary is smoothed to the curved (hyper)surface in $\mathbb{R}^d$ it is supposed to represent. For example, in the case of a boundary that consists of the grid points of a horizontal line in $\mathbb{R}^2$, the real medial axis consists of the vertical lines with odd-half-integer x coordinates. The following definition avoids these unwanted points.

DEFINITION 4 *Let $E = \{e \in \mathbb{Z}^d \mid ||e|| = 1\}$. The* integer medial axis *IMA consists of the points $p \in \mathbb{Z}^d$ such that for some $e \in E$ we have $||ft(p + e) - ft(p)|| > 1$ and $||m - ft(p + e)|| \leq ||m - ft(p)||$ where $m = p + \frac{1}{2}e$ is the midpoint of the line segment from p to $p + e$.*

The second condition on the pair $(p, p+e)$ in the definition of *IMA* is introduced to get one point, rather than two, and specifically the point that is closest to the perpendicular bisector of the line segment from *ft*(p) to *ft*$(p+e)$. If p and $p+e$ have equal claims, both are included. The reason to use *ft* rather than *FT* is that *ft* is computationally cheaper, but also that the restriction of *FT* to $\mathbb{Z}^d$ may well be everywhere single-valued, so that consideration of neighbouring points is needed in any case.

We prefer *IMA* over *CMD* since it is easier to compute and seems to give more image information when the boundary is a discretization of a continuous boundary.

The following lemma is easy to prove.

LEMMA 5 *IMA $\cap B = \emptyset$.*

The definition of *IMA* is primarily motivated by the next result that shows that *IMA* has "enough" elements.

THEOREM 6 *Let p and q be points of the boundary B. Every Manhattan-shortest path from p to q that is not contained in B, contains a point of IMA.*

Proof: Let $r(i)$, $0 \leq i \leq k$ be a Manhattan-shortest path from p to q that is not contained in B. Since it is a Manhattan-shortest path from p to q, we have

$r(0) = p$, $r(k) = q$, and $||r(i+1) - r(i)|| = 1$ for all $0 \leq i < k$. Since the path is not contained in B, there is an index j with $0 < j < k$ and $r(j) \notin B$. Without loss of generality, we may assume $r(1) \notin B$.

Let $x(i) = \mathit{ft}(r(i))$ for all i. Then $x(0) = p$ and $x(k) = q$ and $x(1) \neq r(1)$. We have $||p - r(1)|| = 1$ and hence $\mathit{dt}(r(1)) = 1$. By Lemma 3, this implies that $x(1) = x(0)$ or $||x(1) - x(0)|| > 1$. It follows that function x represents a path from p to q in k steps that is not a Manhattan-shortest path. This implies that there is an index j with $0 \leq j < k$ and $||x(j+1) - x(j)|| > 1$. Put $m = \frac{1}{2}(r(j+1) + r(j))$. If $||m - x(j+1)|| \leq ||m - x(j)||$ then $r(j) \in \mathit{IMA}$. Otherwise $r(j+1) \in \mathit{IMA}$. In that case $j + 1 < k$ because of Lemma 5. □

While the previous result can be interpreted as saying that *IMA* has enough elements, the next result shows that *IMA* has not too many elements, in the sense that every one of them is close to *RMA*.

THEOREM 7 *For every $p \in \mathit{IMA}$, there is $e \in E$ and $t \in \mathbb{R}$ with $0 \leq t \leq \frac{1}{2}$ and $p + te \in \mathit{RMA}$.*

Proof: Let $p \in \mathit{IMA}$. Then there is $e \in E$ with $||\mathit{ft}(p) - \mathit{ft}(p+e)|| > 1$ and $||m - \mathit{ft}(p)|| \geq ||m - \mathit{ft}(p+e)||$ where $m = p + \frac{1}{2}e$. First, assume that $\mathit{ft}(p) \in \mathit{FT}(m)$. Then $\mathit{ft}(p)$ is a closest point on B to m. So $||m - \mathit{ft}(p)|| \leq ||m - \mathit{ft}(p+e)||$. Since $||m - \mathit{ft}(p)|| \geq ||m - \mathit{ft}(p+e)||$, it follows that $||m - \mathit{ft}(p)|| = ||m - \mathit{ft}(p+e)||$ and that both $\mathit{ft}(p)$ and $\mathit{ft}(p+e)$ are elements of $\mathit{FT}(m)$. In view of lemma 2, this implies that $m \in \mathit{RMA}$ is a point as looked for. It remains to treat the case with $\mathit{ft}(p) \notin \mathit{FT}(m)$. Let point z be the last point of the line segment from p to m with $\mathit{ft}(p) \in \mathit{FT}(z)$. By continuity, this point exists. Since $\mathit{ft}(p) \notin \mathit{FT}(z')$ for points z' arbitrary close to z, the set $\mathit{FT}(z)$ consists of more than one element. So $z \in \mathit{RMA}$. □

As illustrated by Theorem 6, *IMA* has some good connectivity properties. In that respect, it is better than *CMD*.

EXAMPLE 8 *Let B be the intersection of $\mathbb{Z}^2$ with the union of the x-axis and the y-axis. Then IMA consists of the points $(x, \pm x)$ for all $x \in \mathbb{Z} \setminus \{0\}$, and CMD is a subset of IMA that contains $(\pm 3, \pm 3)$ and $(\pm 4, \pm 4)$ but misses at least the points $(\pm 1, \pm 1)$, $(\pm 2, \pm 2)$, $(\pm 5, \pm 5)$.*

In general, it seems that, if the complement of B is bounded and connected, then *IMA* is connected (with respect to 8-connectivity in $\mathbb{Z}^2$, or more generally, L_∞-connectivity for $\mathbb{Z}^d$).

A disadvantage of *IMA* is that it can (weakly) depend on the choice of function *ft* within *FT*.

Implementation. The code for the skeletonization step is shown in Fig. 3. One may work with squared distances instead of distances, which avoids the computation of square roots and thus saves time.

When the medial axis is used for image analysis, it is often useful to prune it of disturbing details in some postprocessing phase. Our construction of the integer medial axis yields some information that is very useful for this purpose. The easiest pruning is to strengthen the condition $||ft(p) - ft(p+e)|| > 1$ in the definition of *IMA* by replacing '> 1' by '$> \gamma$' for some pruning parameter γ. This removes some points of *IMA* that are due to irregularities of the boundary.

With the tunable parameter γ, skeletons may be computed according to a user's need. Unwanted skeleton points which still remain can be removed in a postprocessing step, if desired.

```
procedure IMA skeleton
for i := 0 to m − 1 do
  for j := 0 to n − 1 do
    for k := 0 to p − 1 do
      if i > 0 then compare(i,j,k,i-1,j,k) endif
      if j > 0 then compare(i,j,k,i,j-1,k) endif
      if k > 0 then compare(i,j,k,i,j,k-1) endif
    end for
  end for
end for
```

```
procedure compare(i,j,k,p,q,r)
x := [i, j, k]; y := [p, q, r]
x_f := ft_3[x]; y_f := ft_3[y]
if ||x_f − y_f|| > γ then
  crit := inprod(x_f − y_f, x_f + y_f − x − y)
  if crit ≥ 0 then skel[x]:= 1
  endif
  if crit ≤ 0 then skel[y]:= 1
  endif
endif
```

Figure 3. Program fragment for computing the IMA skeleton from the feature transform.

Table 1. Timing results (in seconds) for several data sets.

Data	Size	Feature transform	Skeleton	Total
angio	256x256x128	3	4	7
engine	256x256x128	4	4	8
tooth	256x256x256	7	7	14
vessels	256x256x256	10	6	16
head	256x256x256	9	7	16

4. Results

We have run the skeletonization algorithm on several 3D data sets. Timing results are given for three 3D data sets, i.e. CT scans of a head, a tooth and a number of blood vessels. The size of these sets and their timing results are given in Table 1. These results were obtained on an 1.7 GHz Pentium M processor with 1024 Mb internal memory.

Since the 3D skeletons form surfaces, they are somewhat hard to visualize. Therefore, to get an idea of the quality of our skeletonization algorithm, we first give a number of examples of 2D skeletons, see Fig. 4. For the 3D case, some insight into the structure of the skeleton surfaces can be gained by using volume rendering techniques. An example for the tooth data set is given in Fig. 5. For a better impression a sequence of views from different viewpoints is desired, which can be played as a movie.

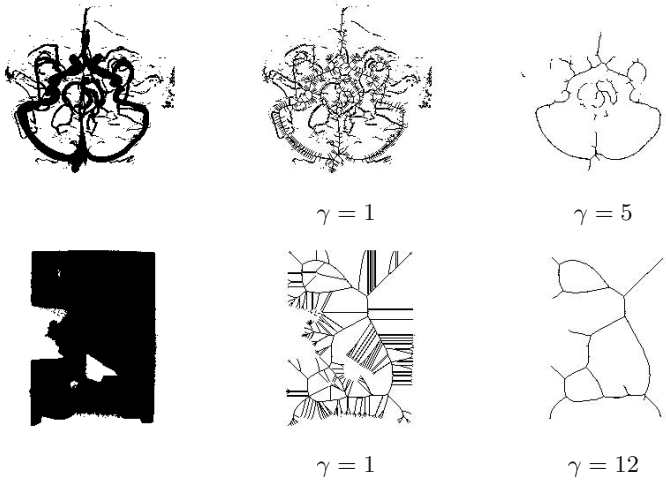
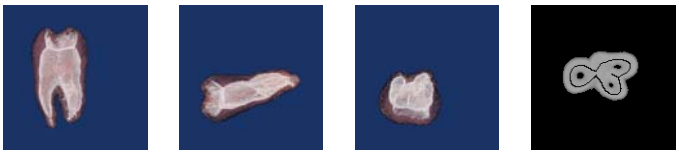


Figure 4. 2D images with their skeletons. Left: original images. Middle: IMA skeleton. Right: pruned IMA skeleton.



(a) top (b) side (c) front (d) slice

Figure 5. (a)-(c): Volume renderings of skeletons (white) inside the original data volumes. (d): Slice of the original tooth data combined with the skeleton.

References

[1] D. Attali and A. Montanvert. Computing and simplifying 2D and 3D continuous skeletons. *Computer Vision and Image Understanding*, 67(3):161–273, 1997.

[2] S. Beucher. Digital skeletons in Euclidean and geodesic spaces. *Signal Processing*, 38:127–141, 1994.

[3] H. Blum. A transformation for extracting new descriptors of shape. In W. Wathen-Dunn, editor, *Proc. Symposium Models for the perception of speech and visual form, Boston, November 1964*, pages 362–380. MIT Press, Cambridge, MA, 1967.

[4] G. Borgefors, I. Nystrom, and G. S. D. Baja. Computing skeletons in three dimensions. *Pattern Recognition*, 32(7):1225–1236, 1999.

[5] D. Coeurjolly. d-Dimensional reverse Euclidean distance transformation and Euclidean medial axis extraction in optimal time. In N. et al., editor, *DGCI 2003*, pages 327–337, New York, 2003. Springer. (LNCS 2886).

[6] P. E. Danielsson. Euclidean distance mapping. *Comp. Graph. Im. Proc.*, 14:227–248, 1980.

[7] Y. Ge and J. Fitzpatrick. On the generation of skeletons from discrete Euclidean distance maps. *IEEE Trans. Pattern Anal. Machine Intell.*, 18:1055–1066, 1996.

[8] T. Hirata. A unified linear-time algorithm for computing distance maps. *Information Processing Letters*, 58:129–133, 1996.

[9] C. M. Ma and M. Sonka. A fully parallel 3d thinning algorithm and its applications. *Computer Vision and Image Understanding*, 64:420–433, 1996.

[10] P. Maragos and R. W. Schafer. Morphological skeleton representation and coding of binary images. *IEEE Trans. Acoust. Speech Signal Proc.*, ASSP-34:1228–1244, 1986.

[11] C. R. Maurer Jr., R. Qi, and V. Raghavan. A linear time algorithm for computing the euclidean distance transform in arbitrary dimensions. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(2):265–270, 2003.

[12] C. R. Maurer Jr., V. Raghavan, and R. Qi. A linear time algorithm for computing the euclidean distance transform in arbitrary dimensions. In *Information Processing in Medical Imaging*, pages 358–364, 2001.

[13] A. Meijster, J. B. T. M. Roerdink, and W. H. Hesselink. A general algorithm for computing distance transforms in linear time. In J. Goutsias, L. Vincent, and D. S. Bloomberg, editors, *Mathematical Morphology and its Applications to Image and Signal Processing*, pages 331–340. Kluwer Acad. Publ., Dordrecht, 2000.

[14] F. Meyer. The binary skeleton in three steps. In *Proc. IEEE Workshop on Computer Architecture and Image Database Management, IEEE Computer Society Press*, pages 477–483, 1985.

[15] J. R. Parker. *Algorithms for Image Processing and Computer Vision*. John Willey & Sons, 1996.

[16] E. Remy and E. Thiel. Look-up tables for medial axis on squared Euclidean distance transform. In N. et al., editor, *DGCI 2003*, pages 224–235, New York, 2003. Springer. (LNCS 2886).

[17] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, New York, 1982.

[18] F. Y. Shih and C. C. Pu. A skeletonization algorithm by maxima tracking on Euclidean distance transform. *Pattern Recognition*, 28(3):331–341, 1995.

[19] H. Talbot and L. Vincent. Euclidean skeletons and conditional bisectors. In *Proc. SPIE Visual Communications and Image Processing'92, Boston (MA)*, volume 1818, pages 862–876, Nov. 1992.

[20] L. Vincent. Efficient computation of various types of skeletons. In *Proc. SPIE Symposium Medical Imaging V, San Jose, CA*, volume 1445, pages 297–311, Feb. 1991.