On the Use of Shape Primitives for Reversible Surface Skeletonization

Stina Svensson[1] and Pieter P. Jonker[2]

[1] Centre for Image Analysis, Swedish University of Agricultural Sciences, Uppsala, Sweden
stina@cb.uu.se
[2] Pattern Recognition Group, Faculty of Applied Sciences Delft University of Technology, Delft, The Netherlands
pieter@ph.tn.tudelft.nl

Abstract. We use a mathematical morphology approach to compute the surface and curve skeletons of a 3D object. We focus on the behaviour of the surface skeleton, in particular the reversibility for the case when the skeleton is, and is not anchored to the set of centres of maximal balls. We elaborate on the difficulties to obtain a reversible surface skeleton that does not depend on the orientation of the original object with respect to the grid, and that has no jagged borders.

Keywords: Topological erosion, mathematical morphology, distance transform.

1 Introduction

For efficient shape analysis of the foreground set in an image, various shape representation schemes have been developed, among which skeletonization is commonly used. Skeletonization is a way to reduce the intrinsic dimension of the foreground objects, i.e., a surface in 2D is reduced to a curve (the 2D skeleton), or a volume in 3D is reduced to a curved surface (the 3D surface skeleton) that might be further reduced to a space curve (the 3D curve skeleton). In this paper we focus on the behaviour of the surface skeleton. To function as an efficient representation scheme, the skeleton of the object should fulfil a number of properties: The skeleton should be a thin subset of the object. To reflect the main structure of the object, the reduction process should not alter the topology. The reduced set should be centred within the original object. The result should be identical under rotation of the original. Finally, the process should be reversible, meaning that the object can be recovered from the skeleton. This property is useful if shape analysis related to changes in thickness of the object is performed. Various approaches to compute the surface skeleton of the foreground set in a 3D image can be found in literature, [1,2,3,4,5,6]. The interest of this paper deals with the latter two, [5,6]. In [5], an algorithm based on conditional erosion of the foreground set was presented. The conditions take care of the preservation of the topology and contain subsets that preserve surfaces,

I. Nyström et al. (Eds.): DGCI 2003, LNCS 2886, pp. 317–326, 2003.

surface ends, curves, curve ends and single points. When iteratively eroding the foreground, they take care that surfaces, curves and single points are not eroded. The resulting skeleton is thin and is centred within the respective object, however, it is not fully reversible. In [6], a distance transform based algorithm to compute reversible surface skeletons was presented. The algorithm is based on topology preserving iterative thinning guided by the distance transform of the foreground set. During this process, voxels needed for reversibility are preserved in combination with voxels needed for surface preservation. The resulting surface skeletons are reversible and centred within the respective objects with respect to the distance function used. However, the surface preservation condition is not enough to avoid a certain jaggedness in the border of the resulting surface skeleton. As far as we know there is to date no reversible surface skeletonization algorithm without this drawback.

In this paper, we study the possibility of constructing an algorithm to compute reversible surface skeletons without this drawback. We show results from an algorithm based on the D^{26} distance, i.e., the 3D equivalence of chess board distance, based on the combination of the algorithms in [5,6]. Moreover, we show to what extent the surface skeletons computed by the algorithm in [5] are reversible and conclude with some remarks on the difficulties in computing reversible surface skeletons that are orientation independent and have no jagged surface borders. See figures 5..8.

2 Notions

Consider a 3D image consisting of foreground X and its complement X^c, the background. In a $3 \times 3 \times 3$ set of voxels centred on a voxel v, there are three types of neighbours to v: 6 face neighbours (on Euclidean distance 1 voxel from v), 12 edge neighbours ($\sqrt{2}$ from v), and 8 vertex neighbours ($\sqrt{3}$ from v). The different neighbours give three types of neighbourhoods, which will be denoted N^6, N^{18}, and N^{26}. A set $A \in X$ is n-connected, $n \in \{6, 18, 26\}$, if each pair of voxels $v_1, v_m \in A$ can be joined by a path $\langle v_1, v_2 \ldots, v_{m-i}, v_m \rangle$ such that each successive pair $\langle v_i, v_{i+1} \rangle$ are n-connected to each other, i.e., $v_i \in N^n(v_{i+1})$. Most often, the highest connectivity is used for the foreground set and the lowest for the background. We adopt this.

Each voxel in an image can be labelled with a distance, according to the chosen distance function, to its closest voxel in the background. The result can be stored in a distance image or distance transform (DT). The DT can be computed in two scans of the images using local distance information only. For more information on how to use and compute DTs, we refer to [7,8,9,10]. We will use a distance function where the distance between two voxels, v and w, is dependent on the number of steps in the face (A), edge (B), and vertex (C) directions in a minimal path between v and w. The distance is given by $d(v, w) = max(A, B, C)$, i.e., the distance equals the number of steps in a minimal 26-connected path between v and w. We refer to this distance as D^{26} and to the corresponding distance transform as the D^{26} DT. The D^{26} DT has the drawback of being unstable under rotation, i.e., it is not a good approximation of the Euclidean DT, [11,12], on

the other hand, the Euclidean DT has the disadvantage in being more difficult to use [13,10]. The label of a voxel v in a DT can be interpreted as the radius of a ball centred on v which is fully enclosed in the foreground (*the DTball*). The foreground can be efficiently represented by a subset of its voxels considering the fact that some of the DTballs are completely covered by other DTballs. A DTball that is not completely covered by another DTball is called a *maximal ball* and the voxel it is centred on a *centre of maximal ball* (CMB). The foreground can be recovered from its CMBs by taking the union of the corresponding DTballs. This process can be efficiently implemented using the reverse distance transformation [14]. The CMBs can be detected on the DT by simple label comparison based on the fact that distance information is not propagated by any CMB to its neighbours. In fact, v, in a D^{26} DT, is a centre of a maximal ball if it has no neighbour with larger distance label. Detecting CMBs on the Euclidean DT is not equally trivial [13]. As far as we know there is to date no publication on the detection of centres of maximal Euclidean balls (for 3D images). A basic morphological operation is the *hit-or-miss* transformation, e.g., [15]. This can be described as a point-by-point transformation of a set X with the structuring element S consisting of two sets S^1 and S^2 and is performed in such a way that $x \in X$ belongs to the transformation Y if and only if S^1_x, i.e., S^1 centred on x, is included in X and S^2_x, i.e., S^2 centred on x, in X^c.

$$Y \leftarrow X \otimes S \equiv \left\{x | S^1_x \subset X, S^2_x \subset X^c\right\}$$

An image is a set of elements (pixels, voxels,...) with underlying vector-space, with vectors k, that represent the positions of the voxels within the image. In an image, we have the foreground set of elements (value 1) and the background set of elements (value 0). The foreground set is constituted by all objects in the image. For a structuring element S, we can associate with the set S^1 foreground elements and with the set S^2 background elements. These are all *do care* elements. To use a structuring element of a fixed size and shape, e.g., to operate upon 3^n neighbourhoods M_k, centred around element x_k in an nD image X, we adopt the notion of *don't care* elements. The transformation $Y \leftarrow X \otimes S$ can be implemented with the neighbourhood transformation $\{\forall k : y_k \leftarrow M_k \cong S\}$. The structuring element S can be used to perform operations like the erosion. The erosion operation on an image X by a structuring element S, $\varepsilon_S(X)$, is equal to all voxels $x \in X$ such that $S_x \in X$, $\varepsilon_S(X) = \{x | S_x \subseteq X\}$. The structuring element S can also be used to put constraints upon the erosion, such as in topology preserving erosion (denoted *topological erosion*) in which only simple elements are eroded. A voxel v belonging to the foreground of image X is called simple if X is homotopic to $X \setminus \{v\}$. Topological erosion is used, e.g., in skeletonization.

3 Shape Primitives

A voxel and its $3 \times 3 \times 3$ neighbourhood comes in four states. It can be part of a single voxel object, it can be a part of a space curve, it can be a part of a curved surface, or it can be part of a volume. As such it can be assigned

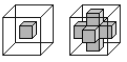

Fig. 1. Structuring elements for conditions for $\tilde{N} = 0$ and $\tilde{N} = 3$, respectively. Foreground voxels are shown in light grey and background voxels are transparent.

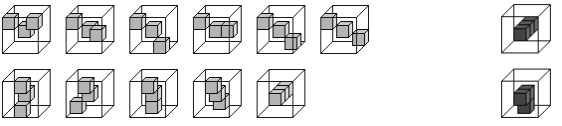

Fig. 2. Curve primitives. Foreground voxels are shown in light grey and background voxels in dark grey.

an object dimension or intrinsic dimension $\tilde{N}$ with $\tilde{N} = 0..3$, respectively. For topological erosion to obtain a surface skeleton, only a volume voxel can be changed from foreground to background if it is on the object boundary, i.e., it has a face neighbour to the background. Changing a foreground surface voxel to background would cause (locally) the creation of a tunnel [16], or, expressed differently, a foreground surface is "pierced" by a background curve. Changing a foreground curve voxel to background would cause (locally) the breaking of one foreground component into two foreground components, or, expressed differently, a foreground curve is "sliced" into two curve parts by a background surface. Changing an isolated foreground voxel to background would cause removal of that foreground component. For $\tilde{N} = 0$ and $\tilde{N} = 3$, topological erosion can be obtained using the structuring elements in Fig. 1 as conditions. To deal with $\tilde{N} = 1$ and $\tilde{N} = 2$, we use the concept of *shape primitives*. A detailed description can be found in [5]. Shape primitives for curves, i.e., $\tilde{N} = 1$, are given by the voxel v and two of its neighbours u and w. These two neighbours should be disconnected. Considering that 26-connectedness is used for the foreground and 6-connectedness for the background, we have for a foreground curve primitive that $u, w \in N^{26}(v)$ and $u \notin N^{26}(w)$ and for a background curve primitive that $u, w \in N^{6}(v)$ and $u \notin N^{6}(w)$. Curve primitives are shown in Fig. 2 (rotated and mirrored primitives are not shown). Shape primitives for a surface, i.e., $\tilde{N} = 2$, can be generated by encircling the voxel v by a simply connected curve, i.e., each curve voxel has exactly two neighbours in the curve. For a foreground surface primitive, the curve is a set of n voxels u_i, where $n \geq 4$ and $u_i \in N^{26}(u_{i+1})$. This, together with $u_i \neq v$, necessarily gives $u_i \in N^{18}(v)$, which effectively expresses that surfaces are locally 18-connected. For a background surface primitive, the curve is a set of at least n voxels u_i, where $n \geq 6$ and $u_i \in N^{6}(u_{i+1})$. This, together with $u_i \neq v$, necessarily gives $u_i \in N^{18}(v)$. Surface primitives are shown in Fig. 3 (rotated and mirrored primitives are not shown).

4 Surface Skeletonization

The shape primitives described in the previous section can be used to find the structuring elements (or, simply, masks) for the conditional erosion to be used

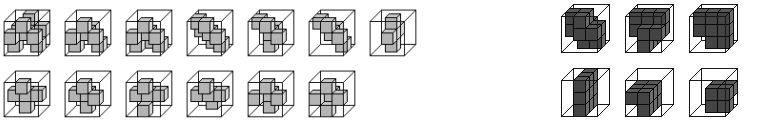

Fig. 3. Surface primitives. Foreground voxels are shown in light grey and background voxels in dark grey.

during skeletonization. To generate the needed masks, curve and surface primitives are suitably intersected. In fact, all possible combinations of intersection of foreground surface primitives and background curve primitives as well as all possible intersections of foreground curve primitives and background surface primitives should be investigated. The masks that will actually be used are those for which a foreground surface is prevented from being penetrated by a background curve (thus, avoiding the creation of a tunnel) and those for which a foreground curve is prevented from being sliced by a background surface (thus, avoiding breaking a foreground component into two components). The masks can be viewed in Fig. 4. For their details, see [5].

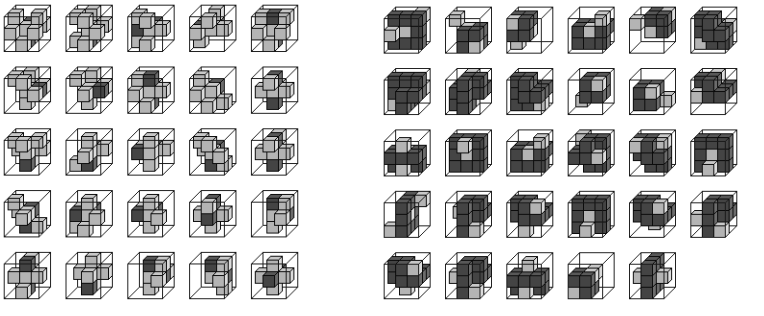

Fig. 4. Mask set for topological erosion with respect to surfaces, left, and curves, right. Foreground voxels are shown in light grey, background voxels in dark grey, and *don't care* voxels transparent.

Using iterated conditional erosion on the foreground (with the generated masks) is not enough to guarantee a topology preserving removal of voxels. For example, two-voxel thick parts of the foreground can not be properly detected using a $3 \times 3 \times 3$ neighbourhood. This can be solved based on the use of subiterations, where the foreground is eroded from one direction only in each subiteration; or a subfield sequential method, where the image is examined in a directional and sequential fashion [4]. Yet another approach is to use a recursive neighbourhood [17]. In this case, masks are applied both simultaneously and sequentially, and each structuring element is matched both in the input and in the output image, where only voxels from the current iteration are considered. The recursive neighbourhood method is the fastest procedure [17].

The shape primitives are also used to generate surface and curve end conditions [5]. A curve extends from a voxel v in two directions. The curve end conditions are found by setting one of the two neighbours of v to background. A surface extends from a voxel v in four directions. Systematically one and two directions can be set to background, yielding half and quarter surfaces.

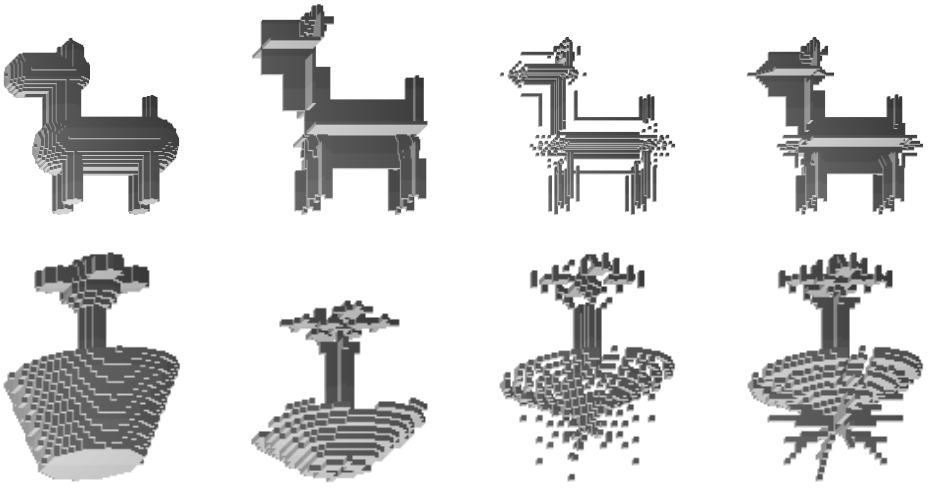

Fig. 5. From left to right: Object, non-reversible surface skeleton, centres of maximal D^{26} balls, and D^{26} surface skeleton.

The first step in the skeletonization algorithm is to compute the D^{26} DT, from which the CMBs are extracted. Anchoring the surface skeleton onto the CMBs will give a reversible skeletonization. For the actual skeletonization, the process is the same as in [5], except for two details. We use distance guided erosion, meaning that for each erosion iteration, we only consider voxels with a distance label equal to the iteration number. CMBs are never removed. We denote this skeleton the D^{26} surface skeleton. For details on the implementation, we refer to [18]. In Fig. 5, we show the surface skeletons for two objects as well as the set of CMBs and the surface skeleton obtained when anchoring to the CMBs is omitted. The latter is denoted non-reversible surface skeleton. The examples are "dog" and "pot plant".

5 Reversibility and Surface Preservation

The reversible surface skeletons, using the anchoring to the CMBs, are reversible with respect to the D^{26} distance: the object can be fully recovered from the skeleton, e.g., by applying the reverse D^{26} DT. If the anchoring in CMBs is omitted, the resulting surface skeleton will not be reversible. The surface skeletonization algorithm described in [5], from which this distance guided algorithm originates,

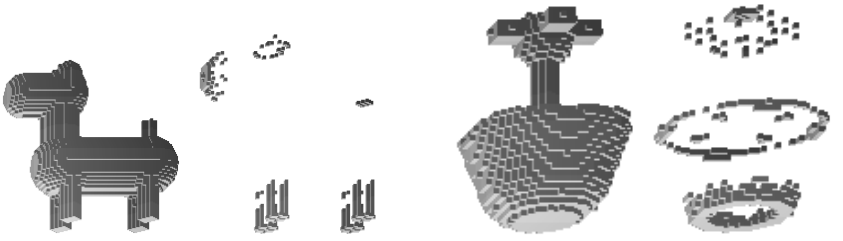

Fig. 6. Recovered objects when anchoring is omitted, left. Difference with respect to the original objects, right.

aims to compute a surface skeleton that is centred within the object with respect to the Euclidean distance, i.e., to give a surface skeleton which is stable under rotation. If we assign distance labels from the Euclidean DT to the surface skeleton and then apply the reverse Euclidean DT (here computed in a brute-force way by, for each skeletal voxel v, adding a Euclidean ball centred on v with radius equal to the distance label of v), the objects in Fig. 6 are obtained. Comparing to the original objects, we have recovered 16903 of the 17229 voxels for the "dog" and 8413 of the 8871 for the "pot plant". If we consider a more brick-like object, the situation is slightly worse, see Fig. 7. Compared to the original objects, we have recovered 259475 of the 283604 voxels for the "pyramid" and 260658 of the 288040 for the "rotated pyramid". Observe that the surface skeletons differ in shape for different rotations. This is due to the problem of perfect alignment of objects made of square voxels in a square tessellated grid. Possibly, special surface-end preservation conditions can be made that detect the situations in which the object is perfectly aligned to the grid, and hence could make the top row of Fig. 7 similar to the bottom row. However, for this probably a larger support of the surface end-preservation conditions is necessary, e.g., 5^3, but such a patch may also have as drawback that it preserves voxels in non-aligned cases, where this is not preferable. This needs further research. In conclusion, anchoring to the CMBs should be used to ensure reversibility, however, this does not ensure that the obtained skeleton is independent of the rotation of the original.

Using anchoring to CMBs also does not guarantee that a surface skeleton without jagged borders is obtained. Fig. 8 shows that the anchoring in CMBs gives dashed lines from the top of the cone to the ground plane. The skeletonization algorithm produces a correct result, however, it anchors to single points in space and not to surface patches: the cone is eroded surface by surface and the CMBs emerge on the gradually eroded surface of the cone leading to single voxel skeleton branches. Hence, the Christmas tree pattern is correct, although it does not look nice. If we also want to preserve jagged surface borders, we can add additional conditions, e.g., as shown in Fig. 9. (They have the same effect as the surface preservation condition in [6]). Note, however, that these masks only preserve battlements where there is only a single background voxel between each pair of foreground voxels. Its effect is a form of surface border closing with a support of one. If there is a distance of two background voxels between each pair

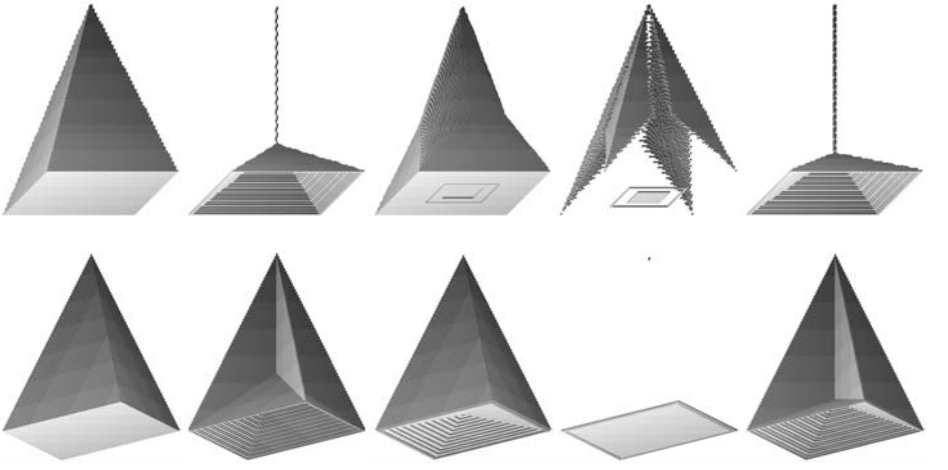

Fig. 7. From left to right: Object, non-reversible surface skeleton, recovered object, difference with respect to the original object, and D^{26} surface skeleton. (The pyramid in the bottom row is a rotated version (45° around y) of the pyramid in the top row.)

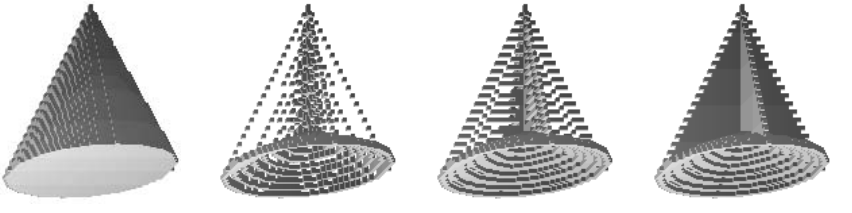

Fig. 8. From left to right: Object, centres of maximal D^{26} balls, D^{26} surface skeleton, and improved D^{26} surface skeleton.

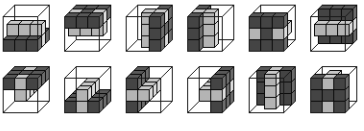

Fig. 9. Additional masks intended for surface shape preservation. Foreground voxels are shown in light grey, background voxels in dark grey, and *don't care* voxels transparent.

of foreground voxels, larger support, e.g., 5^3 needs to be taken. But this leads to a scale problem; how large should the support be? The actual cause is in the fact that we want to obtain a certain surface boundary property (that involves a direction), but we anchor to points (that have no direction). Further research has to be done on, e.g., using the centres of maximum ellipses, which give anchors that are line-pieces. This probably will lead to smooth surface skeleton borders, as then the surface end-conditions will match on them, in contrast with the current situation, where only the curve end-conditions match on the anchor points.

6 Conclusion

In this paper, we have pointed out the difficulties in finding an algorithm for computing a reversible surface skeleton of a 3D object which results in a surface skeleton that is also independent of the orientation of the original object with respect to the grid and has no jagged surface borders. We showed that surface skeletons resulting from the algorithm described in [5] are fairly stable under rotation and that the original object could be largely recovered. For full reversibility, anchoring to centres of maximal balls is needed. We have shown the results both for a distance guided algorithm derived from [5] and based on the D^{26} DT, leading to the conclusion that we can make fully reversible surface skeletons that are centred within the original object with respect to a D^{26} distance. But also that even if we combine [5] with anchoring on centres of maximal Euclidean balls, the problems with orientation and jagged surface borders remain. To better preserve jagged surface borders, conditions that perform a closing operation on the skeleton border in addition to the topology preservation conditions, can be used to improve the result. However, the orientation problem due to perfect alignment of objects and masks, made of square voxels on a square tessellated grid, remains. In real applications the orientation dependence and the jagged surface skeleton border are minor problems. From a theoretical point, however, it would be interesting to develop an algorithm, for which the skeletal properties of orientation independence and smooth surface skeleton boundaries are guaranteed.

Acknowledgement

The work presented in this paper was carried out while Stina Svensson visited the Pattern Recognition Group in Delft, The Netherlands. This was made possible by a grant from The Swedish Foundation for International Cooperation in Research and Higher Education (STINT).

References

1. Attali, D., Lachaud, J.O.: Delaunay conforming iso-surface, skeleton extraction and noise removal. Computational Geometry **19** (2001) 175–189
2. Leymarie, F.F., Kimia, B.B.: The shock scaffold for representing 3D shape. In Arcelli, C., Cordella, L.P., Sanniti di Baja, G., eds.: Visual Form 2001. Volume 2059 of Lecture Notes in Computer Science., Capri, Italy, Springer-Verlag (2001) 216–228
3. Ma, C.M., Wan, S.Y.: A medial-surface oriented 3-d two-subfield thinning algorithm. Pattern Recognition Letters **22** (2001) 1439–1446
4. Palágyi, K., Kuba, A.: A parallel 3D 12-subiteration thinning algorithm. Graphical Models and Image Processing **61** (1999) 199–221
5. Jonker, P.P.: Skeletons in N dimensions using shape primitives. Pattern Recognition Letters **23** (2002) 677–686

6. Svensson, S.: Reversible surface skeletons of 3D objects by iterative thinning of distance transforms. In Bertrand, G., Imiya, A., Klette, R., eds.: Digital and Image Geometry. Volume 2243 of Lecture Notes in Computer Science., Dagstuhl, Germany, Springer-Verlag (2002) 395–406
7. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. Journal of the Association for Computing Machinery **13** (1966) 471–494
8. Borgefors, G.: Applications using distance transforms. In Arcelli, C., Cordella, L.P., Sanniti di Baja, G., eds.: Aspects of Visual Form Processing. World Scientific Publishing Co. Pte. Ltd. (1994) 83–108
9. Borgefors, G.: On digital distance transforms in three dimensions. Computer Vision and Image Understanding **64** (1996) 368–376
10. Svensson, S., Borgefors, G.: Digital distance transforms in 3D images using information from neighbourhoods up to $5 \times 5 \times 5$. Computer Vision and Image Understanding **88** (2002) 24–53
11. Danielsson, P.E.: Euclidean distance mapping. Computer Graphics and Image Processing **14** (1980) 227–248
12. Ragnemalm, I.: The Euclidean distance transform in arbitrary dimensions. Pattern Recognition Letters **14** (1993) 883–888
13. Borgefors, G., Ragnemalm, I., Sanniti di Baja, G.: The Euclidean distance transform: Finding the local maxima and reconstructing the shape. In Johansen, P., Olsen, S., eds.: Proceedings of Scandinavian Conference on Image Analysis (SCIA'91), Pattern Recognition Society of Denmark (1991) 974–981
14. Nyström, I., Borgefors, G.: Synthesising objects and scenes using the reverse distance transformation in 2D and 3D. In Braccini, C., Floriani, L.D., Vernazza, G., eds.: Proceedings of ICIAP'95: Image Analysis and Processing, Springer-Verlag (1995) 441–446
15. Serra, J.: Image Analysis and Mathematical Morphology. Academic Pres, inc. (1982)
16. Kong, T.Y.: A digital fundamental group. Computers & Graphics **13** (1989) 159–166
17. Jonker, P.P.: Morphological operations in recursive neighbourhoods. Accepted for publication in Pattern Recognition Letters (2002)
18. Jonker, P.P.: Lecture notes on mathematical morphology for 2, 3 & 4 dimensional images and its implementation in soft and hardware. In Wojciechowski, K., ed.: Summer School on Mathematical Morphology and Signal Processing. Volume 2., Zakopane, Poland (1995) 41–120 ISBN 83-904743-2-8.