PERGAMON

Pattern Recognition 32 (1999) 1225–1236

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

Computing skeletons in three dimensions

Gunilla Borgefors[a,*], Ingela Nyström[b], Gabriella Sanniti Di Baja[c]

[a]*Centre for Image Analysis, Swedish University of Agricultural Sciences, SE-752 37 Uppsala, Sweden*
[b]*Centre for Image Analysis, Uppsala University, SE-752 37 Uppsala, Sweden*
[c]*Istituto di Cibernetica, National Research Council of Italy, IT-800 72 Arco Felice (Napoli), Italy*

Received 29 April 1997; received in revised form 18 May 1998

Abstract

Skeletonization will probably become as valuable a tool for shape analysis in 3D, as it is in 2D. We present a topology preserving 3D skeletonization method which computes both surface and curve skeletons whose voxels are labelled with the D^6 distance to the original background. The surface skeleton preserves all shape information, so (close to) complete recovery of the object is possible. The curve skeleton preserves the general geometry of the object. No complex computations, large sets of masks, or extra memory are used, which make implementations efficient. Resulting skeletons for geometric objects in a number of 2 Mbyte images are shown as examples.

Keywords: Volume image; Shape representation; Surface skeleton; Curve skeleton; Thinning; Digital topology

1. Introduction

The history of skeletonization of digital objects is almost as old as digital image analysis itself. The purpose is to reduce 2D discrete objects to (1D) linear representations preserving topological and geometrical information. Given an input binary image, skeletonization changes non-skeletal object pixels into background pixels. Regardless of the scheme adopted to perform skeletonization, the resulting skeleton is a union of curves placed symmetrically with respect to the border of the object. The literature on 2D skeletonization is very rich. An outline of various thinning and skeletonization methodologies can be found in Ref. [1].

Reducing discrete structures to lower dimensions is even more desirable when dealing with (3D) volume images. The result of 3D skeletonization is either a set of 3D surfaces and curves, or if even more compression is desired, a set of only 3D curves. The compression to 3D curves is possible only for solid objects having no cavities. A hollow torus, e.g. could never be reasonably represented by a curve skeleton. The skeleton could be a promising tool for an increasing number of applications, especially in biomedical imagery. However, compared to the literature on 2D skeletonization, the articles published on 3D skeletonization are still not very numerous. The main reason for this seems to be the difficulty to address and efficiently solve essential problems, such as topology preservation, in more than two dimensions. In fact, although many concepts such as Euler characteristics, simple points and connectivity have been studied in the past years (e.g. Refs. [2–6]), the implementation of skeletonization methods based on their use is rather complicated.

The general strategy for 3D skeletonization does not differ significantly from the strategy in the 2D case.

*Corresponding author. Tel.: 00 46 18 471 3466; fax: 00 46 18 553 447; e-mail: junilla@cb.u.se

0031-3203/99/$20.00
PII: S0031-3203(98)00082-X

Object voxels are changed to background voxels under the constraint that topology and geometry of the object are preserved. However, a number of new problems have to be solved in the 3D case. For example, when designing topology preserving removal operations, besides preventing disconnections and creation of cavities, which must be done also in 2D, one must also avoid the creation of tunnels and the excavation of unwanted deep cavities in complex surfaces. It seems that the geometry preserving criteria, i.e. the definition of protrusions or end-points, are what separates existing algorithms; every author has his own criteria, resulting in very different skeletons.

If the skeletal voxels are labelled with their distance to the original background and skeletonization is performed in a way that guarantees the inclusion in the skeleton of object voxels having locally maximal distance, skeletonization becomes reversible. The object can be recovered by applying the reverse distance transformation to the skeleton, see Ref. [7]. This recovery property is disregarded in many approaches, but is, e.g. present in an algorithm to compute surface skeletons that we have recently proposed [8]. A curve skeleton cannot include enough information for recovery in the general case.

An early work on curve skeletons can be found in Ref. [9], where iterative thinning is performed based on (unfortunately not sufficient) conditions for preservation of the Euler characteristics. In Ref. [10], a surface skeleton is obtained for 6-connected objects using topological numbers in an algorithm using six directional sub-iterations. A thinning scheme based upon eight sub-fields can be found in Ref. [11], where (only) 26-connectivity is preserved, either for surface or curve skeletons using the same topological numbers. In Ref. [12], iterative erosion resulting in surface skeletons is performed on the object border using (rather complex) contour information. In Ref. [5], simple points (removable voxels) are identified using (fairly large) look-up tables, and removed in three-scan iterations. The algorithm preserves topology and is insensitive to noise, but geometry is not fully preserved by the resulting surface skeletons. This work is continued in Ref. [13]. In Ref. [4], the Euler characteristics are stored in look-up tables and used to identify simple points in a six directional sub-iterations algorithm. The result is either a surface or a curve skeleton depending on which end-point condition is used. In Ref. [14], six directional sub-iterations are also used. Nice rotation independent surface or curve skeletons are obtained by using the Euclidean distance transform. In Ref. [15], objects are reduced directly to 3D curves using four classes (each with a number) of deleting templates. The algorithm is demonstrated on visual analysis of computer tomography lung data. In many other papers on skeletonization of volume objects, the only examples given are tiny test images, which makes it difficult to understand what the results would be for reasonably sized and/or real images.

The concept of skeletonization has also been extended to four-dimensional data, e.g. Ref. [16], where the Euler characteristics are utilized, and different end-point conditions decide the dimensionality of the skeleton.

In this paper we describe a method for thinning a volume (voxel) object to a skeleton whose voxels are labelled with the (D^6) distance to the original background. Our skeletonization method is performed in two major steps. The first step reduces the object to a *surface skeleton* (Section 3), which requires two iterative phases. During the first phase, non-multiple voxels are iteratively removed until an at most two voxel thick surface of skeletal voxels is identified. During the second phase, this set is reduced to a set of one voxel thin surfaces (and curves). The original object can be recovered from its surface skeleton, using the distance labels. The recovery is exact if started from the "thick" skeleton from the first phase, whereas some border voxels may be missing if the one voxel thin skeleton is used. The second step reduces the surface skeleton to a *curve skeleton* (Section 4), which also requires two phases. The skeletons are topology preserving, but some (exceptional) objects cannot be reduced to skeletons. We present resulting skeletons for a number of 2 Mbyte images (Section 5). The voxels of our curve skeletons are labelled with the distance to the original background. This is useful information also for the curve skeletons, even though the original object cannot be recovered in this case.

2. Definitions

Each voxel v has three types of neighbours among its 26 closest neighbours; 6 face-, 12 edge-, and 8 point-neighbours, that share a face, an edge, and a point with v, respectively.

In a binary image we define an object component as a 26-connected set of voxels. As a consequence of the 26-connectedness selected for the object, 6-connectedness must be used for the background. If the background consists of exactly one 6-connected component, then the object is termed a *solid* object.

A *border* voxel is an object voxel with a face-neighbour in the background. Object voxels, which are not border voxels, are *internal* voxels.

Voxels are characterized by the numbers of n-connected components of object or background in their neighbourhood. For example, a voxel is a *break-point* voxel if it has more than one 26-connected object component in its 26-neighbourhood. The recursive algorithm in Ref. [17] can be used to count connected components efficiently.

N_f18 is defined as the number of 6-connected background components in the 18-neighbourhood of a voxel having the central voxel as a face-neighbour. On a 3D surface, an *outer* voxel is a surface voxel with $N_f18 = 1$. If $N_f18 > 1$ it is called a *tunnel* voxel, because removing it

would create a tunnel through the object. This classification of surface voxels is discussed in Refs. [18] and [19].

The D^6 metric is obtained by counting the number of steps in the minimal 6-connected path between voxels. The D^6 distance is the 3D equivalent of the city-block distance in 2D, see Ref. [20].

The algorithm in this paper is partly parallel in nature, partly sequential. Operations performed in parallel are parallel in the sense that the local operations are parallel, i.e. all voxels can and should be processed simultaneously in each iteration. In sequential operations, neighbouring voxels cannot be processed simultaneously. Our method is implemented on standard sequential computers, using pseudo-parallel programs.

3. Surface skeletonization

The algorithm described in the following is an improved version of the one presented in Ref. [8].

3.1. Identification of removable voxels

Voxels are iteratively removed from the object, until no more voxels can be removed. Each iteration consists of four steps (or scans through the image). Every step can be performed in parallel.

Step 1:

(1) Among voxels not already labelled, identify border voxels, and label them with the current iteration number.

(2) Among internal voxels, identify voxels with an edge-neighbour in the background and label them with the current iteration number plus one.

Step 2: Among voxels labelled with the current iteration number, mark those that are multiple.

Step 3: Among non-multiple voxels labelled with the current iteration number, mark as tunnel voxels those for which $N_f 18 > 1$. When computing $N_f 18 > 1$ neighbouring non-multiple voxels labelled with the current iteration number are interpreted as background voxels, i.e. they are treated as if they were already removed.

Step 4: Remove all unmarked border voxels.

The process terminates when all border voxels are multiple, and hence no more voxels can be removed.

During Step 1 border voxels in all directions are simultaneously identified, in contrast to some earlier parallel thinning work, e.g. Refs. [4,10], where each iteration is divided into six directional sub-iterations. That directional strategy is a generalization of the 2D case, but it may cause topological problems in 3D, such as disconnecting objects.

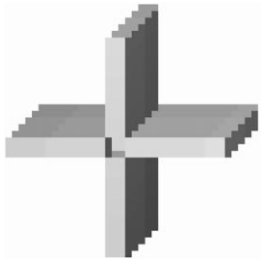

Fig. 1. Where two surfaces cross the outermost voxels of the crossing are removed, but creation of deep cavities is prevented by Step 1 (ii) of the algorithm.

Since only border voxels are candidates for removal, cavities are not created. Labelling the voxels with an edge-neighbour in the background with the iteration number plus one, gives them the correct distance label (from the original background). This is important for object recovery, and guarantees that Step 2 will be performed on all voxels with the same distance in the same iteration. An illustrative example is shown using two crossing planes, see Fig. 1. Consider the internal voxels placed at the intersection of the planes, and suppose that we do not label them as requested by Step 1 (ii). Removal of the border voxels in the intersection, exposes to the background other voxels in the intersection. These voxels would become removable in the next iteration. By iterating this removal, the surface skeleton would finally have four planes linked to each other only by a single middle voxel. This would still be topologically correct, but not very geometry preserving. If we instead consider that all voxels in the intersection would have the *same* distance label in the D^6 distance transform of the object, it becomes apparent that they should all be checked for removal in the *same* iteration. Labelling the voxels with an edge-neighbour in the background as in Step 1 (ii) prevents creation of deep, narrow cavities where two surfaces intersect. Consequently, only the outermost voxels of the crossings are removed.

In Step 2 we identify multiple voxels in a way that is a generalization of a 2D algorithm that identifies the multiple pixels on the border of an 8-connected object, see Ref. [21]. Multiple pixels have been proved to be equivalent to pixels that can not be removed during skeletonization, see Ref. [22]. The resulting skeletal pixels are labelled as they would be in the city-block distance transform of the object. Moreover, it can be shown that the multiple pixels include all the pixels that are local maxima of the city-block distance transform.

For volume images, we define a border voxel of the current iteration, v, as multiple if any of Conditions A1–A3 is satisfied:

*Condition A*1: No pair of opposite face-neighbours (aligned along one of the three principal planes) of v exists

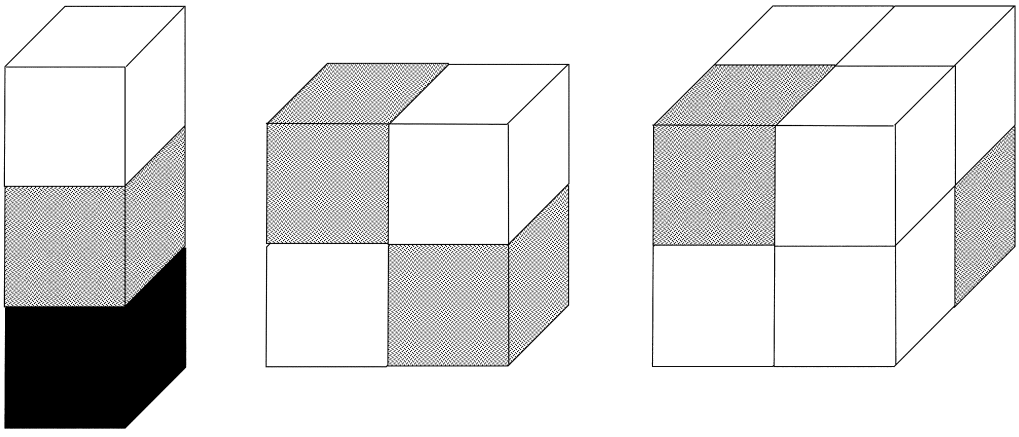

Fig. 2. The voxel configurations in Conditions A1–A3. Internal voxels are black, border voxels are grey, and background voxels are white. Rotations of the configurations result in six cases for Condition A1 (left), twelve cases for Condition A2 (middle), and eight cases for Condition A3 (right).

such that one of them is an internal voxel and the other a background voxel.

*Condition A*2: A 2×2 neighbourhood of v (in any of the three principal planes) exists such that the edge-neighbour of v is a border voxel (of the current or an earlier iteration), and the two face-neighbours are background voxels.

*Condition A*3: A $2 \times 2 \times 2$ neighbourhood of v exists such that the point-neighbour of v is a border voxel (of the current or an earlier iteration), while the other six neighbours are background voxels.

These conditions are illustrated in Fig. 2. Condition A1 means that the configuration to the left in the figure does *not* occur in any of the three principal planes. Condition A2 means that the configuration in the middle, or any of its rotations, occurs and Condition A3 means that the configuration to the right, or any of its rotations, occurs. Condition A1 guarantees that protrusion voxels, including the tip (end-point), are not removed. All three conditions prevent disconnecting the object. Condition A3 is unique to the 3D case and is necessary as the object is defined as 26-connected.

Step 3 is necessary to prevent tunnel creation in thin complex objects, see Refs. [18, 19]. This problem is even more pronounced when reducing the surfaces to curves, and will be described in detail in Section 4. At first, it may seem as if Steps 2 and 3 can be performed in one single step. However, they cannot; all multiple voxels have to be identified before the consequence of removal of non-multiple voxels is known.

The skeletal voxels found by this algorithm constitute a 26-connected set of voxels, which is at most two voxels thick. The voxels are labelled with the iteration number, which means that they have the same label as they would have if the D^6 distance transformation had been applied to the object. The definition of a border voxel ensures that each successive border layer is 6-connected to the previous layer, exactly as the layers are ordered in the D^6 distance transform. For very "noisy" objects, whose borders consists entirely of multiple voxels, the algorithm will not work (compare with "Arcelli sets" in 2D [23]). These objects can generally be created only artificially. If our skeletonization method was applied to such objects, no voxels would be removed and the resulting "skeleton" would be more than two voxels thick. We will not consider this case further. Due to Condition A1, the skeletal voxels include all the voxels that are local maxima of the D^6 distance transform of the object. A local maximum is defined as a voxel with a label larger than or equal to that of its face-neighbours. None of the face-neighbours of a local maximum can then be an internal voxel (an internal voxel would be labelled with the next iteration number!) and thus Condition A1 holds for the local maximum.

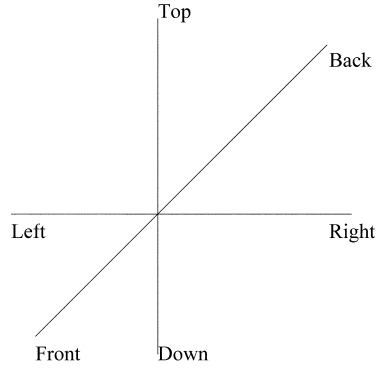

Fig. 3. Directions in voxel space.

A few internal voxels might still remain after this first thinning in regions where many surfaces and/or curves meet (due to the fact that remaining object voxels have formed "Arcelli sets"). It is even conceivable that these voxels are not yet labelled, i.e. they do not have an edge-neighbour in the background. Such voxels should be assigned a label equal to the minimum label in their 6-neighbourhood plus one, which corresponds to the value they would have in the D^6 distance transform.

As local maxima are not removed, object recovery is possible, using the *reverse* D^6 distance transformation, see Ref. [7]. The pseudo-code for the forward pass is given below. The backward pass is similar, but scans through the image in the opposite direction.

```
/*----------FORWARD PASS----------*/
LOOP_X_Y_Z (image) {
  if (x < xLO || x > xHI || y < yLO || y > yHI
     || z < zLO || z > zHI)
  image = 0; /* image border omitted */
  else
  image = MAX( I(x, y, z − 1) − 1,
               I(x, y − 1, z) − 1,
               I(x − 1, y, z) − 1,
               I )
}
/*------END--FORWARD PASS---------*/
```

3.2. Thinning to unit-wide surface skeleton

The skeletal set obtained is at most two voxels thick. It can be reduced to unit thickness by applying a thinning process that has to be split into six directional processes, each of them applied once. Using directional processes is necessary to prevent breaking connectedness and excessive shortening.

The six processes occur sequentially in the directions Top–Down, Down–Top, Left–Right, Right–Left, Front–Back, and Back–Front, see Fig. 3. A Top-voxel is defined as a skeletal voxel with a background voxel as the face-neighbour in the Top-direction. The other five are similarly defined.

In the Top–Down process Top-voxels are candidates for sequential removal. A Top-voxel v is removed if all Conditions B1–B3 are satisfied:

*Condition B*1: The face-neighbour of v in the Down-direction is a Down-voxel. See Fig. 4.
*Condition B*2: Condition A2 does not occur.
*Condition B*3: Condition A3 does not occur.

Condition B1 guarantees that the current Top–voxel belongs to a portion of the set of the skeletal voxels which is *exactly* two voxels thick in the Top-Down direction. For example, see Fig. 5, illustrating a set that is two voxels thick in the Front–Back direction. The Top-voxels of the set do not satisfy Condition B1, so they are not removed during the Top–Down process, thus, unwanted "shrinking" of the set is prevented in the Top–Down direction.

Condition B2 prevents breaking the connectedness of the set when the connection occurs between edge-neighbours only. Only four of the twelve rotational cases of Condition A2 (see middle of Fig. 2) have to be checked, as Condition B1 guarantees connection in the other

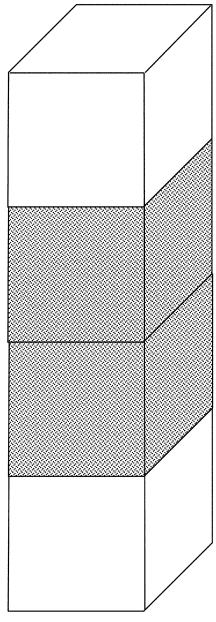

Fig. 4. Four-voxels configuration for removing Top-voxels (see text). Border voxels are grey, background voxels are white.

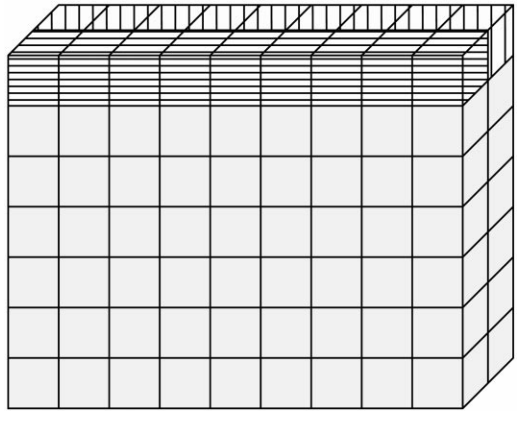

Fig. 5. A voxel set, two voxels thick in the Front–Back direction. The Top-voxels (hatched) are not removed in the Top–Down process, but the horizontally hatched Top-voxels are also Front-voxels and will be removed during the Front-Back process.

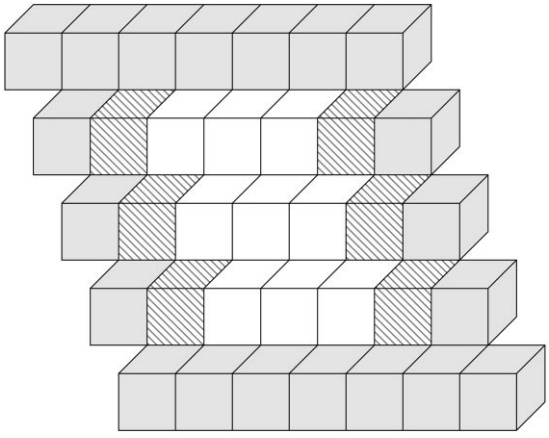

Fig. 6. A thin surface where the outer voxels are marked in grey. The hatched voxels are erroneously identified as outer voxels, if counting the 6-connected background components in the 26-neighbourhood; the 18-neighbourhood should used.

cases. The cases to be checked are those where the edge-neighbour connects to one of the four upper edges of the voxel.

Condition B3 prevents breaking the connectedness of the set when the connection occurs between point-neighbours only. As Condition B1 guarantees connection in four cases, Condition A3 has to be checked only for the cases where the point-neighbour connects to one of the four upper corners of the voxel (see right of Fig. 2).

The other five processes are similar. After the six directional processes have been applied, the skeleton consists of thin surfaces and curves. Some of the local maxima of the implied distance transform could have been removed during this thinning process, so complete object recovery by the reverse distance transformation is no longer possible. However, the voxels that are not recovered are all border voxels of the original object, so the distortion of object shape is not great.

4. Curve skeletonization

The surface skeleton can be further reduced to a curve skeleton. The original object cannot thereafter be recovered, but its topology and geometry are preserved. The skeletal voxels are labelled with their (D^6) distance to the original background, which might be useful information in quantitative analysis of the shape.

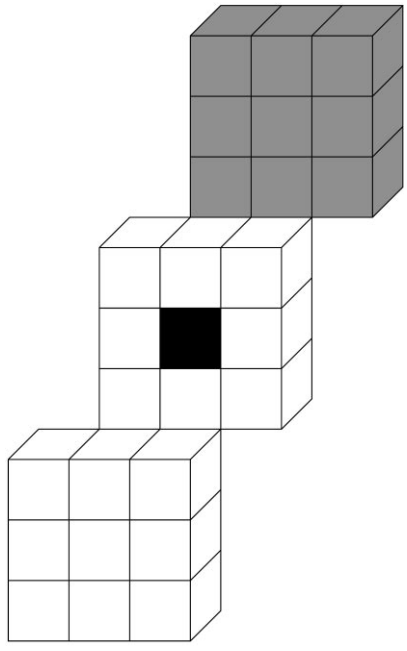

Fig. 7. A 26-neighbourhood shown slice with a "cap" in the Front-direction. The central voxel is a tip-of-protection voxel. Background voxels are white. "Don't care" voxels are grey. Rotation of the "cap" gives the other five direction masks.

4.1. Identification of removable voxels

Voxels are iteratively removed from the surface, until no more voxels can be removed. Each iteration consists of three steps (or scans through the image). Differently from before, only the two first steps are performed in parallel.

Step 1: Identify outer voxels, $N_f 18 = 1$, on the surface.

Step 2: Inspect all outer voxels (both from the current and earlier iterations). Mark a voxel as removable, if it

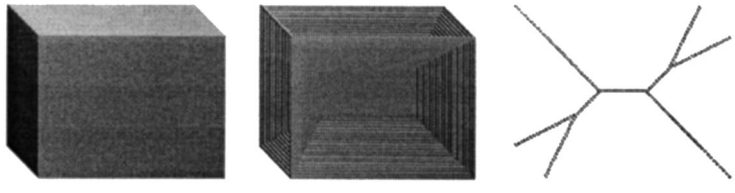

Fig. 8. A box (left), its surface skeleton of labelled voxels (middle), and its curve skeleton (right).

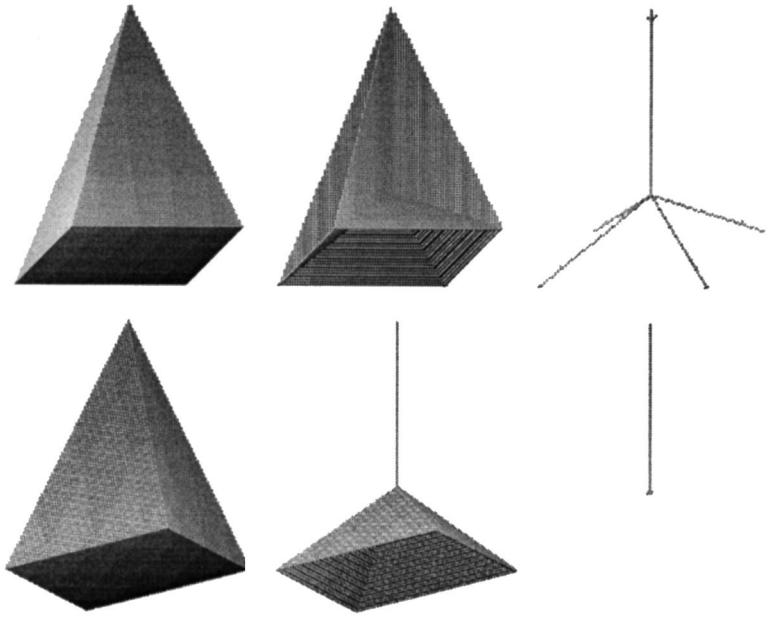

Fig. 9. Top: A pyramid (left), its surface skeleton (middle), and its curve skeleton (right). Bottom: The pyramid rotated 45° (left), its surface skeleton (middle), and its curve skeleton (right).

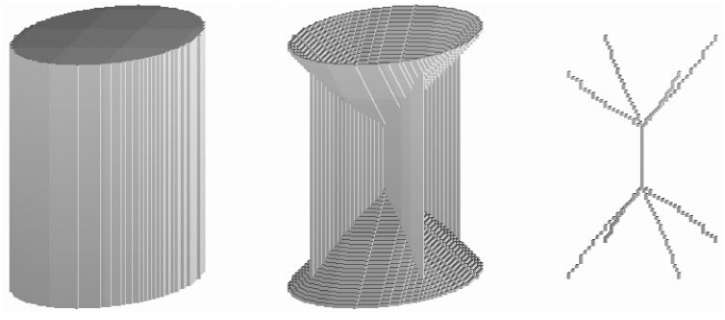

Fig. 10. A cylinder (left), its surface skeleton (middle), and its curve skeleton (right).

has two 26-connected components of outer voxels, but only one 26-connected component of object in its 26-neighbourhood.

Step 3: Sequentially remove marked voxels, unless they are break-point voxels.

During Step 1 the voxels which are candidates for removal, the outer voxels, are identified. Intuitively, interior surface voxels have more than one background component in their neighbourhood, and voxels on the border of a surface have one background component. As the background is 6-connected, its components are counted using 6-connectedness. In Fig. 6 the outer voxels of a surface are marked in grey. When counting components in the 26-neighbourhood, the hatched voxels are also identified as outer voxels, as the background is connected through the point-neighbours of these voxels. Removal of a hatched voxel would change the topology of the object as a tunnel would be created, therefore they must not be identified as outer voxels. A 3D surface is actually 18-connected, therefore the 18-neighbourhood should be used when counting background components. Simply counting components in the 18-neighbourhood will not however, identify all removable voxels. For some complex surfaces the number of background components for removable border voxels are more than one, as background voxels being edge-connected to the central voxel are included as components (of size one voxel). The solution is to count only the 6-connected background

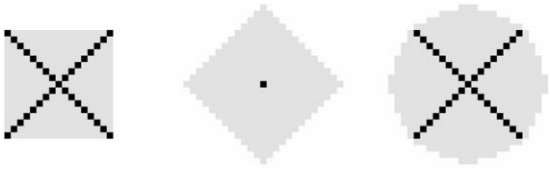

Fig. 11. Three 2D objects and their skeletons imposed in black. A square (left), a rhomb (middle), and a digital Euclidean circle (right).

components having the central voxel as a face-neighbour, thus $N_f 18$. Our definition of outer voxels relates to the definitions in Refs. [18,19].

During Step 2 every (current and earlier) outer voxel is checked for removal. The voxel must have only one 26-connected object component; otherwise, the object will be disconnected. Also, it must have two components of outer voxels; otherwise, it may be the tip of a protrusion. As an example, imagine a flat square surface with the outer voxels marked. Along the sides there are two components of outer voxels, while the four corner voxels only have one such 26-connected component and hence are not removable. The resulting skeleton is X-shaped, which is analogous to what happens in 2D skeletonization based on the city-block distance. A consequence of this protrusion preservation is that, for complex surfaces, some unwanted voxels may remain, giving the curve skeleton a "cloudy" look. This happens when the outer voxels do not form a connected set, and hence some outer voxels only have one component of outer voxels, even though they do not belong to a protrusion. This "cloudiness" can be taken care of in post-processing. Outer voxels identified in earlier iterations are checked in subsequent iterations as the removal of neighbouring voxels may create two components of outer voxels, which then allows removal.

The removal of any single voxel marked in Step 2 preserves connectedness, but the simultaneous removal of all of them may disconnect the object. Therefore, Step 3 must be sequential. Before removing a voxel the number of 26-connected object components in its 26-neighbourhood are counted. If the number of components is greater than one, the voxel must remain as the object otherwise would be disconnected.

4.2. Thinning to unit-wide curve skeleton

In case the original surface is (locally) an even number of voxels broad, the curve skeleton becomes two voxels wide. It can be reduced to unit thickness by iteratively applying a final thinning process, until no more voxels can be removed. Each iteration consists of three steps (or scans through the image). Only the two first steps are performed in parallel.

Step 1: Classify *tip-of-protrusion* voxels, for which a "cap" of background voxels fits in any of the Top-, Down-, Left-, Right-, Front-, or Back-directions (see Fig. 7). Classify *break-point* voxels, which have more than one component of object in their 26-neighbourhood.

Step 2: Among non-classified voxels, mark outer voxels, $N_f 18 = 1$, whose removal will not create tunnels.

Step 3: Sequentially remove the outer voxels, unless the object is disconnected.

The result is a 26-connected unit-wide curve skeleton with the topology and geometry of the original object preserved.

5. Examples

We are well aware of the difficulty in interpreting the projections of our thin 3D skeletons. A preferable way to visualize this kind of images is a rotation sequence of projections. We assure the reader, that all skeletons *are* connected, without tunnels, and one voxel thin, even though the figures could sometimes be misleading in this respect.

Volume objects were synthesized in $128 \times 128 \times 128$ images. A first example of the results of our skeletonization method is illustrated in Fig. 8. A box of size $40 \times 60 \times 80$ voxels is shown to the left. The surface skeleton of the box can be seen in the middle. The skeleton consists of 5.4% of the original voxels, and includes

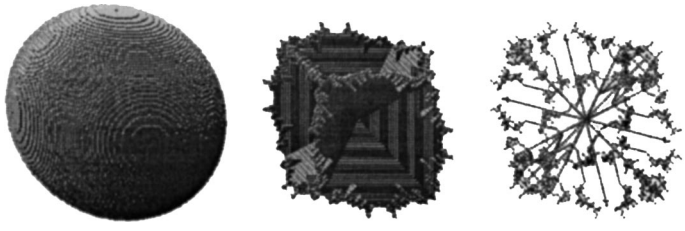

Fig. 12. A digital Euclidean sphere with radius 50 voxels (left), its surface skeleton (middle), and curve skeleton (right).

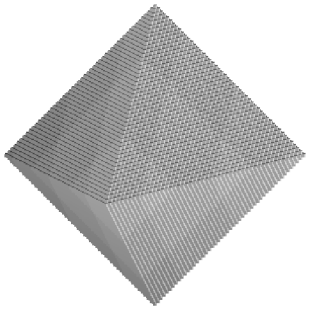

Fig. 13. A D^6 sphere, i.e. an octahedron, with radius 50 voxels (left), its surface skeleton (middle), and curve skeleton (right).

(almost) all the information present in the original image. The original object can be recovered, except for some of the original border voxels. The curve skeleton of the box can be seen to the right, describing the shape of the original object well.

The results of skeletonization of a pyramid (base 85×85 and height 122 voxels), the same pyramid rotated 45°, and a cylinder (radius 30 voxels and height 80 voxels), can be seen in Figs. 9 and 10, respectively.

When looking at these results, remember that the underlying metric is the D^6 one. Comparison can be made to the 2D thinning algorithm in reference (21), where the underlying metric is the city-block metric. The skeletons for some 2D objects can be seen in Fig. 11. Compare these with the pyramids in Fig. 9, and the cylinder in Fig. 10.

The surface and curve skeletons of a digital Euclidean "sphere" are rather complex. In the digital world, this object is a complex one, especially if the D^6 metric is assumed, as it is here. A simple object, on the other hand, is the octahedron, which is the D^6 (or face-connectivity) sphere. The surface and curve skeletons of the octahedron are single voxels. Skeletons for the two spheres are shown in Figs. 12 and 13. A skeleton based on the Euclidean metric would give a simple skeleton for the Euclidean sphere, but a complex one for the octahedron.

Elongated objects are well suited for reduction to curves, see Fig. 14, where an object consisting of three fused cylinders has been reduced to surface and curve skeletons.

The computations for the 2 Mbyte images in Figs. 8–10 and 12–14 took in total (surface *and* curve skeletonization), on the average, 80 CPU seconds on an ordinary (DEC Alpha) workstation.

6. Discussion and conclusion

In this paper we present a skeletonization method for volume objects, which computes both surface and curve skeletons. The surface skeleton preserves all shape information, so that (close to) complete recovery of the object is possible. The curve skeleton preserves the general geometry of the object. No disconnections, cavities, or tunnels are created. Objects with cavities cannot be reduced to curve skeletons. Some surfaces will remain in our "curve" skeleton in these cases. If they did not, the skeletonization algorithm would not be topology preserving.

The surface and curve skeletonization methods are iterative in nature, but, in contrast to many other approaches, the whole outer layer of voxels is treated simultaneously so that there are no directional sub-iterations. This, and the fact that no complex computations, large sets of masks, or extra memory are used, makes the implementation efficient. In both cases a final thinning

Fig. 14. Three fused cyliders (left), its surface skeleton (middle), and curve skeleton (right).

step is necessary, as the first skeleton will be two voxels thick where the original object has even thickness.

As the underlying metric used in the skeletonization is D^6, the skeletons produced suffers from the same weakness as the skeletonization based on the city-block metric in 2D (see Ref. [20]); the skeletons are not rotation independent, as illustrated by the synthetic object in Fig. 9. Objects in real application images do not accentuate the problem like this, though. Real objects are not likely to be smooth, and, hence, the skeletons are complex. Their extra or missing skeleton branches due to rotation can often be disregarded. The rotation dependendcy problem can be alleviated by a pre-processing step, where the object is rotated into a "normalized" position, for example, by placing the maximum diameter along one coordinate axis, the maximum thickness orthogonal to this axis as another coordinate axis, and finally the axis orthogonal to these two as the third coordinate axis, see Ref. [24]. In this way, similar structures will always get the same normalized orientation. A future solution is to develop skeletons based on more rotation-independent underlying metrics. Our skeletonization can be seen as the starting point to understand how skeletons can be extracted from any distance map.

Every protrusion that is not (locally) a corner of an octahedron will generate a skeletal branch, i.e., skeletonization is sensitive to noise. Pre-processing in the form of morphological smoothing operations will alleviate (but not solve) this problem, especially if a small octahedron is used as a structuring element. A simple way to implement smoothing in our algorithm, is to apply unconditioned removal for a number of iterations proportional to the expected noise size (with the drawback that significant skeleton branches are correspondingly shortened). Realistically, however, skeletons from real applications should be pruned as both significant and unwanted skeletal branches will be present in our skeletons. Hence, the pruning task is to identify and remove the unwanted branches. Methods with pruning built into the iterative thinning process might not be able to distinguish between unwanted and significant branches, when not using information from the total skeleton, and therefore remove some significant branches. Developing good pruning strategies for 3D surface and curve skeletons is an important problem for further research. As for the skeletonization itself, the pruning can be expected to be significantly more complex than in 2D.

Because of the underlying metric, objects with flat surfaces will produce much "nicer" skeletons than objects with curved surfaces (cf. Figs. 8 and 12). The only way to solve this is to develop skeletons based on more rotation independent underlying metrics.

Skeletonization has proved to be a valuable tool for shape analysis in 2D. We have no doubt that it will eventually prove so also in 3D. So far, the major part of volume images have been representations of various parts of the human body. Skeletonization of different organs within the body will make manipulation and analysis of their shape easier. For compact objects, such as the kidneys or liver, surface skeletonization is suitable. For tube-like objects, such as blood vessels and trachea, curve skeletonization preserves the essential information, especially since the curve voxels are marked with the current diameter of the tube. Various 3D imaging techniques are becoming more and more available, and so are the necessary memory and computing power to handle these images. This means that many new applications will appear. Production quality control, both in macro and micro scales, for industrial products (e.g. paper, cloth, and machine parts), animal, and vegetable "objects" and tissues (e.g. fruit, seed, and cell structure), comes to mind.

7. Summary-computing skeletons in three dimensions

Skeletonization of (3D) volume objects denotes either reduction to a 2D structure of 3D surfaces and curves, or, if even more compression is desired and the object to be skeletonized has no cavities, reduction to a 1D structure of only 3D curves. The general strategy for 3D skeletonization doet not differ significantly from the strategy in the 2D case. Object voxels are changed to background voxels under the constraint that geometry of the object and topology are preserved.

In this paper we present a topology preserving skeletoniztion method for volume objects, which computes both surface and curve skeletons. The surface skeleton preserves all shape information, so that (close to) complete recovery of the object is possible. The curve skeleton preserves the general geometry of the object. Voxels of our skeletons are labelled with the D^6 distance to the original background. This is useful information not only for the surface skeleton where it enables object recovery, but also for the curve skeletons, even though the original object cannot be recovered in this case. No disconnections cavities or tunnels are created.

Our skeletonization method is performed in two major steps. The first step reduces the object to a surface skeleton, which requires two iterative phases. During the first pase, non-multiple voxels are iteratively removed until an at most two voxel thick surface of skeletal voxels is identified. During the second phase, this set is reduced to a set of one voxel thick surfaces (and curves). The original object can be recovered from its surface skeleton, using the distance labels. The second step reduces the surface skeleton, to a curve skeleton, which also requires two phases. The first reduces the surfaces to curves, the second thins the curves to one voxel thickness. We present resulting skeletons for a number of synthetic and real 2 Mbyte images.

The surface and curve skeletonization methods are iterative in nature, but, in contrast to many other

approaches, the whole outer layer of voxels is treated simultaneously, so that there are no directional subiterations. This, and the fact that no complex computations or extra memory are used, makes the implementations efficient, even on an standard sequential workstation, where, on the average, 80 CPU seconds are enough to skeletonize objects in 2 Mbyte images.

Skeletonization has proved to be a valuable tool for shape analysis in 2D. We have no doubt that it will eventually prove so also in 3D. So far, the major part of volume images have been representations of various parts of the human body, but many new applications will appear with the increase in computer capacity. Production quality control, both in macro and micro scales, for industrial products (e.g. paper, cloth, and machine parts), animal and vegetable "objects" and tissues (e.g. fruit, seed, and cell structure), comes to mind.

Acknowledgements

Scientific support were given by Prof. Ewert Bengtsson and Dr. Bo Nordin, which is gratefully acknowledged, as is the financial support of the Swedish Research Council for Engineering Science (TFR), grant number 95–182.

References

[1] L. Lam, S.-W. Lee, C.Y. Suen, Thinning methodologies – a comprehensive survey, IEEE Trans. Pattern Anal. Mach. Intell. 14 (9) (1992) 869–885.

[2] T.Y. Kong, A. Rosenfeld, Digital topology: introduction and survey, Comput. Vision, Graphics, Image Process. 48 (1989) 357–393.

[3] G. Bertrand, Simple points, topological numbers and geodesic neighbourhoods in cubic grids, Pattern Recognition Lett. 15 (1994) 1003–1011.

[4] T.-C. Lee, R.L. Kashyap, C-N. Chu, Building skeleton models via 3-D medial surface/axis thinning algorithms, CVGIP: Graphical Models Image Process. 56 (6) (1994) 462–478.

[5] P.K. Saha, B.B. Chaudhuri. Detection of 3-D simple points for topology preserving transformations with application to thinning, IEEE Trans. on Pattern Anal. Mach. Intell. 16 (10) (1994) 1028–1032.

[6] T.Y. Kong, On topology preservation in 2-D and 3-D thinning, Int. J. Pattern Recognition Artificial Intell. 9 (5) (1995) 813–844.

[7] I. Nyström, G. Borgefors. Synthesising objects and scenes using the reverse distance transformation in 2D and 3D, in: C. Braccini, L. DeFloriani, G. Vernazza, (Eds.), Proceedings of ICIAP'95: Image Analysis and Processing, Springer, Berlin, 1995, pp. 441–446.

[8] G. Borgefors, I. Nyström, G.S. di Baja, Surface skeletonization of volume objects, in: P. Perner, P. Wang, and A. Rosenfeld (Eds.), Proceedings of SSPR'96: Advances in Structural and Syntactical Pattern Recognition, Springer, Berlin, Heidelberg, 1996, pp. 251–259.

[9] S. Lobregt, P.W. Verbeek, F.C.A. Groen, Three-dimensional skeletonization: principle and algorithm, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2 (1) (1980) 75–77.

[10] G. Bertrand, A parallel thinning algorithm for medial surfaces, Pattern Recognition Lett. 16 (1995) 979–986.

[11] G. Bertrand, Z. Aktouf, A three-dimensional thinning algorithm using subfields, in: R.A. Melter, A.Y. Wu (Eds.), Vision Geometry III, Proc. *SPIE* 2356, 1994, pp. 113–124.

[12] S. Miguet, V. Marion-Poty, A new 2-D and 3-D thinning algorithm based on successive border generations, Proc. 4th Conf. on Discrete Geometry in Computer Imagery, Grenoble, France, 1994, pp. 195–206.

[13] P.K. Saha, D.D. Majumder, A topology and shape preserving thinning and segmentation method for 3D digital space, Image Process. Commun. 2 (3) (1997) 3–34.

[14] T. Saito, J.I. Toriwaki, A sequential thinning algorithm for three dimensional digital pictures using the Euclidean distance transformation, Proc. 9th Scand. Conf. on Image Analysis, Uppsala, Sweden, 1995, pp. 507–51.

[15] Cherng Min Ma, M. Sonka, A fully parallel 3D thinning algorithm and its applications. Computer Vision Image Understanding, 64 (3) (1996) 420–433.

[16] P.P. Jonker, O. Vermeij, On skeletonization in 4D images, in: P. Perner, P. Wang, A. Rosenfeld (Eds.), Proc. SSPR'96: Advances in Structural and Syntactical Pattern Recognition, Springer, Berlin, Heidelberg, 1996, pp. 79–89.

[17] G. Borgefors, I. Nyström, G.S. di Baja, Connected components in 3D neighbourhoods, Proc. 10th Scand. Conf. on Image Analysis, Lappeenranta, Finland, 1997, pp. 567–572.

[18] G. Malandain, G. Bertrand, N. Ayache, Topological segmentation of discrete surfaces, Int. J. Comput. Vision 10 (2) (1993) 183–197.

[19] P.K. Saha, B.B. Chaudhuri, 3D digital topology under binary transformation with applications, Comput. Vision Image Understanding 63 (3) (1996) 418–429.

[20] G. Borgefors, On digital distance transforms in three dimensions, Comput. Vision Image Understanding 64 (3) (1996) 368–376.

[21] C. Arcelli, G.S. di Baja. A one- pass two-operations process to detect the skeletal pixels on the 4-distance transform, IEEE Trans. Pattern Anal. Mac. Intell. 11 (4) (1989) 411–414.

[22] C. Arcelli, G.S. di Baja, A contour characterization for multiply connected figures, Pattern Recognition Lett., 6 (1987) 245–249.

[23] C. Arcelli. Pattern thinning by contour tracing, Comput. Graphics Image Process. 17 (1981) 130–144.

[24] I. Nyström, E. Bengtsson, B. Nordin, G. Borgefors, Quantitative analysis of volume images – electron microscopic tomography of HIV, Medical Imaging 1994: Image Processing, Proc. SPIE 2167, 1994, pp. 296–303.

About the Author—GUNILLA BORGEFORS received the M. Eng. and Lic. Eng. in Applied Mathematics, from Linköping University in 1975 and 1983, respectively; her Ph.D. in Numerical Analysis, from the Royal Institute of Technology, Stockholm in 1986; and her "Docent" in Image Processing from Linköping University in 1992, all in Sweden. From 1982 to 1993 she was employed at the National Defence Research Establishment, Linköping, Sweden, eventually as Director of Research for computer vision, and in 1990–1993 as Head of the Division of Information Systems. From 1993 she is full Professor at Centre for Image Analysis, Swedish University of Agricultural Sciences,Uppsala, Sweden. Borgefors was President of the Swedish Society for Automated Image Analysis in 1988–1992 and Secretary and 1st Vice President of the International Association for Pattern Recognition, in 1990–1994 and 1994–96, respectively. Borgefors has published a large number of papers in international journals and conferences, and has been the editor of three books on image analysis. Her current research interests are digital geometry in two, three and higher dimensions and the application of image analysis in remote sensing and in industry.

About the Author—INGELA NYSTRÖM received the M.Sc., degree in Applied Computer Science and Mathematics, and the Ph.D. degree in Computerized Image Analysis from Uppsala University, Sweden, in 1991, and in 1997, respectively. She is currently a Researcher and Lecturer at the Centre for Image Analysis, Uppsala, Sweden. Her research interest is method development for qualitative shape analysis of volume objects.

About the Author—GABRIELLA SANNITI Di Baja received the Doctoral degree in Physics from the University of Naples, Naples, Italy, in 1973. Since then, she has been working in the field of Image Processing and Pattern Recognition at the Istituto di Cibernetica of the National Research Council of Italy, Naples, where she currently has the position of Director of Research. Sanniti di Baja has published more than one hundred papers in international journals and conference proceedings, and has been editor of four books. Her main research activities concern two-dimensional shape representation, decomposition and description. Sanniti di Baja is one of the organizers of the International Workshop on Visual Form. She has been chairman of the Education Committee of the International Association for Pattern Recognition (IAPR) 1991–1994. Since October 1994, she is the IAPR Secretary.