The <3,4,5> Curvilinear Skeleton

Carlo Arcelli, Gabriella Sanniti di Baja, and Luca Serino

Institute of Cybernetics "E. Caianiello", CNR, Via Campi Flegrei 34
80078 Pozzuoli (Naples), Italy
{c.arcelli,g.sannitidibaja,l.serino}@cib.na.cnr.it

Abstract. A new skeletonization algorithm is presented to compute the curvilinear skeleton of 3D objects. The algorithm is based on the use of the <3,4,5> distance transform, on the detection of suitable anchor points, and on iterated topology preserving voxel removal. The obtained skeleton is topologically correct, is symmetrically placed within the object and its structure reflects the morphology of the represented object.

1 Introduction

The curvilinear skeleton of a 3D object is a representation consisting of the union of curves that has received much attention in the literature; a comprehensive list of contributions can be found in two recent books [1,2]. Such a representation can be computed only for objects rid of cavities. In fact, one of the features of the skeleton is its topological equivalence with the object. Thus, for objects including cavities, a topologically equivalent linear representation cannot be obtained, since each cavity should be mapped into a cavity in the resulting set, which implies that a closed surface surrounding each cavity should be found in the representation.

Besides topological equivalence with the object (meaning that the curvilinear skeleton of a connected object has to be connected and should include as many closed curves, as many are the tunnels of the object), the curvilinear skeleton allows to reduce dimensionality (since a 3D object is represented by a union of curves), is symmetrically placed within the object, and each of the curves constituting it can be seen in correspondence with a part of the object. The latter property is particularly of interest in the framework of the structural approach to object analysis and recognition. In fact, the curvilinear skeleton can be split into the constituting curves and each curve can be used to analyze the part of the object it represents. The spatial relations among the curves of the skeleton can be used to derive information on the organization of the parts constituting the object. For example, this approach has been followed in the 2D case, when the skeleton is used in the framework of OCR systems. The character is thinned down to its skeleton, which is then decomposed into parts corresponding to the strokes forming the character. Also in the 3D case, decomposition of the curvilinear skeleton into its constituting curves can be useful for applications, e.g., for deformable object modeling.

One of the most appealing approaches to skeletonization is based on the use of the distance transform, e.g., [3,13]. As it is well known, all object elements in the distance

S. Brlek, C. Reutenauer, and X. Provençal (Eds.): DGCI 2009, LNCS 5810, pp. 409–420, 2009.

transform are assigned the value of their distance from a reference set. In the 2D case, the landscape paradigm can be followed according to which the 2D distance transform image is interpreted as a 3D elevation map, where distance values of pixels in the 2D image denote the elevations of those pixels with respect to the ground level, i.e., the reference set, in the 3D elevation map. The skeleton can be computed by detecting peaks and ridges in the 3D landscape and by suitably connecting them. Skeletons of 2D objects computed in this way are definitely centered within the objects and, if suitable criteria are used to connect ridges and peaks, are topologically correct. Another important feature of these skeletons is that they are reversible. In fact, the peaks and most of the ridges are centers of maximal discs in the distance transform and it is well known that the union of the maximal discs coincides in shape and size with the original object [14].

Centers of maximal balls in a 3D distance transform have the same property characterizing the centers of maximal discs in the 2D distance transform as concerns their envelope. However, except for the case of 3D objects with tubular shape, where the centers of maximal balls are mainly aligned along linear structures, the centers of maximal balls are often organized in patch structures. As a consequence, the inclusion of all centers of maximal balls in the curvilinear skeleton is not compatible with the requested linear structure of the skeleton. Thus, for 3D objects the curvilinear skeleton does not enjoy the reversibility property. However, the curvilinear skeleton is still of interest for applications if its morphological structure reflects sufficiently well the most salient aspects of the object's shape.

In the 3D case, the landscape paradigm could still be followed, by interpreting the 3D distance transform image as an elevation map in four dimensions. This process would be rather complex. In this paper, we prefer to resort to the classical approach, based on iterated topology preserving voxel removal, and to integrate it with criteria for the detection of suitable anchor points in the distance transform, so as to avoid unwanted shortening of skeleton branches.

We point out that also skeletonization methods that do not explicitly refer to the distance transform, e.g., those based on boundary evolution, do use information derived from the distance transform [15-19]. If the boundary of an object is interpreted as moving towards the inside of the object, the boundary motion can be modeled as a continuous wavefront by means of a partial differential equation and the singularities of the flow can be detected to compute the skeleton, by using the gradient of the Euclidean distance to the boundary (or the gradient of an approximation of the Euclidean distance).

Our skeletonization method consists of two phases. During the first phase, the 3D object is transformed into a topologically equivalent set consisting of both line- and patch-structures. During the second phase, such a set is furthermore compressed into the curvilinear skeleton. The distance transform is used both to reduce the computation time and to guarantee that the curvilinear skeleton is centered within the object and adequately represents it. The method is described in Section 2, together with some necessary notions and definitions. Experimental results are discussed in Section 3 and a brief conclusion is given in Section 4.

2 The Curvilinear Skeleton

We deal with binary voxel images in cubic grids, where the object F is the set of 1's and the background B is the set of 0's, and use the 26-connectedness and the 6-connectedness for the object and the background, respectively. The 3×3×3 neighborhood N(p) of a voxel p includes the 6 face-, the 12 edge- and the 8 vertex-neighbors of p (respectively denoted by n_f, n_e, n_v). The set N*(p) includes only the 6 face- and the 12 edge-neighbors of p. The generic neighbor of p is denoted by n_i, where i stands for f, e, or v.

We assume that F does not include any cavity otherwise the curvilinear skeleton could not be computed unless altering topology.

The number of 26-connected object components computed in N(p), and the number of 6-connected background components having p as face-neighbor and computed in N*(p) are respectively denoted by c_p and by c^*_p.

An object voxel p is *simple* if the two objects with and without p are topologically equivalent. In [20,21] it has been shown that p is simple iff c_p=1 and c^*_p=1.

The general scheme of the proposed skeletonization method is the following.

During the first phase, aimed at extracting from the 3D object a subset consisting of patches and lines, the distance transform of the input object from its complement is computed; a proper subset of the set of centers of maximal balls detected in the distance transform is taken as the set of anchor points; object voxels are checked in increasing distance value order and are removed (i.e., are assigned the background value 0) if they are not anchor points and their removal does not alter topology; the obtained set of patches and lines is simplified (i.e., is reduced to unit-thickness and some peripheral patches and lines regarded as non-meaningful are removed).

During the second phase, a classification of the voxels of the obtained set is done; the distance transform of the set is computed; a subset of the set of centers of maximal balls, as well as voxels classified as curve- branching- junction-voxels are taken as anchor points; voxels are checked in increasing distance value order and are removed if they are not anchor points and their removal does not alter topology; finally, the obtained set of lines is simplified (i.e., is reduced to the unit-wide curvilinear skeleton and peripheral branches regarded as non-meaningful are removed).

2.1 First Phase

The distance transform DT_F of F with respect to B is computed by using the <3,4,5> distance function [22], which provides a reasonably good approximation of the Euclidean distance and is particularly suited when dealing with digital objects.

The k-th layer of DT_F is the set including all voxels, whose distance value p satisfies the condition $3(k\text{-}1) < p \leq 3k$, see [23].

The border of the object at the k-th iteration of skeletonization consists of the voxels belonging to previous layers and that have not been removed at previous iterations, as well as of the voxels in the k-th layer. All other object voxels constitute the inside of the object at the k-th iteration.

In DT_F, a center of a maximal ball, CMB_F, is an object voxel p such that for each of its object neighbors n_i it results $n_i<p+w_i$, where w_i is 3, 4 or 5, depending on

whether *i* stands for *f*, *e*, or *v*. Actually, voxels with distance value 3 must be seen as having the *equivalent* distance value 1, in order the above rule always holds [23].

As pointed out in the Introduction, inclusion of all the CMB$_F$s of the 3D object in its curvilinear skeleton is not generally possible. Thus, we perform a selection of the CMB$_F$s to be considered as anchor points, so as to make the set resulting at the end of the first phase easier to be furthermore processed. We note that some CMB$_F$s (termed CMB1$_F$s) either have no object neighbors in the successive more internal layer or have there at most vertex-neighbors, while the remaining CMB$_F$s (termed CMB2$_F$s) have also face- or edge-neighbors in the successive more internal layer. While the CMB1$_F$s are voxels that are equidistant from two opposite parts of the object border, the CMB2$_F$s, especially if sparse, are either due to discretization effects or are found in correspondence with two parts of the object border intersecting each other so as to form a 90 degree or larger dihedral angle. For example, if the original continuous object is a cylinder, some CMB$_F$s can be created in DT$_F$ of the discrete cylinder that do not carry relevant information, since they are not aligned along the main symmetry axis of the cylinder. In fact the circular border delimiting each section of the continuous cylinder is unavoidably transformed into a discrete polygon, which will be reflected by the presence of CMB$_F$s in the distance transform due to the weak convexities along the polygonal border. If the original object is shaped like a brick, besides the CMB$_F$s in the middle of the brick, which are due to opposite faces of the brick, other CMB$_F$s will be found in correspondence with the 90 degree dihedral angles formed by two contiguous faces. The latter CMB$_F$s, if kept as anchor points, would generate in the resulting set some peripheral patches that would only make the second phase of the process more laborious. Thus, we select as anchor points only the CMB1$_F$s.

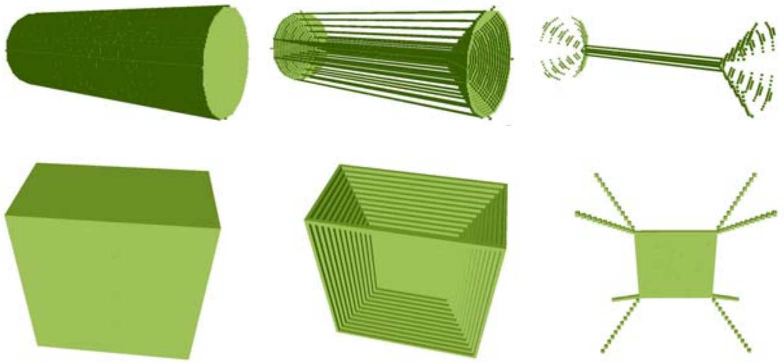

Fig. 1. A cylinder and a brick, left, all their CMB$_F$s, middle, and their CMB1$_F$s, right

Two simple examples are shown in Fig. 1. There, a cylinder and a brick, Fig. 1 left, all their CMB$_F$s, Fig. 1 middle, and the CMB1$_F$s, Fig. 1 right, are shown. Preserving as anchor points all the CMB$_F$s would lead to sets including a number of peripheral patches that should be compressed anyway during the second phase of skeletonization. In turn, selecting as anchor points only the CMB1$_F$s allows us to derive information from the most internal parts of the object and also to keep track of the 90 degree dihedral angles (see the linear subsets of the $CMB1_F$s for the brick, which are due to the triples of contiguous faces), while having a reduced set of patches and lines to handle during the second phase.

The voxels in DT_F are examined starting with those with the minimal distance value 3 in increasing distance order and, if they were not selected as anchor points, topology preserving removal is sequentially accomplished. Only simple voxels are removed. Thus, if p is the current distance value, a voxel p is removed if c_p=1 and c^*_p=1.

After all voxels with distance value p have been checked, before examining voxels with the successive distance value, we perform a process to maintain the superficial structure of those sets of non-removed voxels organized into patches. Any voxel p that has not been removed identifies and marks as non-removable its neighbors, if any, which are the most suitable for linking p to the inside. To this aim, the gradient from each non-removed voxel with distance value p towards each of its neighbors with distance value larger than p is computed. All neighbors for which the gradient results to be maximum are marked as non-removable. The gradient from p to its neighbor n_i is computed as $(n_i–p)/w_i$, where w_i is 3, 4 or 5, depending on whether i stands for f, e, or v.

Once all distance values have been taken into account, a set PL consisting of patches and lines is obtained. PL is nearly-thin, i.e., is at most 2-voxel thick.

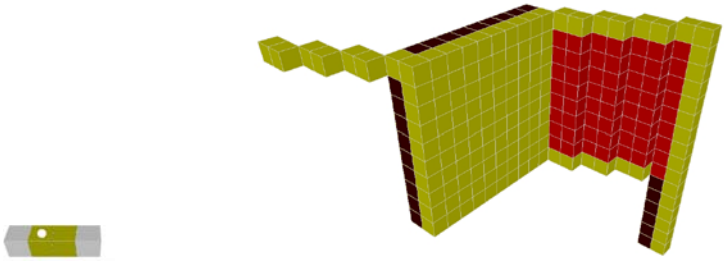

Fig. 2. One of the directional masks to check 2-voxel thickness, left; a set including 1- and 2-voxel thick patches and lines, right

The nearly-thin set PL is reduced to unit-thickness by means of final thinning. To this aim, object voxels are divided into two categories: type1 are the voxels characterized by $c^*_p≠1$, (see dark gray voxels in Fig. 2 right) and type2 are all other object voxels (see light gray and black voxels in Fig. 2 right). Final thinning is concerned only with type2 voxels that are placed in two-voxel thick configurations. Thickening of PL can occur in face- or edge-direction. Thus, directional masks are successively used to identify thick sets of type2 voxels. In Fig. 2 left, one of the masks to detect thickening in face-direction is shown. In the mask, the two internal voxels are type2 voxels, while the external voxels are background voxels. The masks in the remaining face- and edge-directions have a similar structure. Any type2 voxel p in a thick configuration, according to the currently used directional mask, is sequentially removed provided that there exists exactly one 26-connected component of type2 voxels in N(p), and p has more than one neighboring object voxel. These two conditions are adequate to reduce thickness by removing only those voxels (e.g., the black voxels in Fig.2 right) placed in 2-voxel thick patches or lines, while preserving from removal voxels for which the directional mask is satisfied but they are placed along one-voxel thick patches or lines. For the running example shown in Fig. 3 left, the unit-wide set of patches and lines is given in Fig. 3 middle.

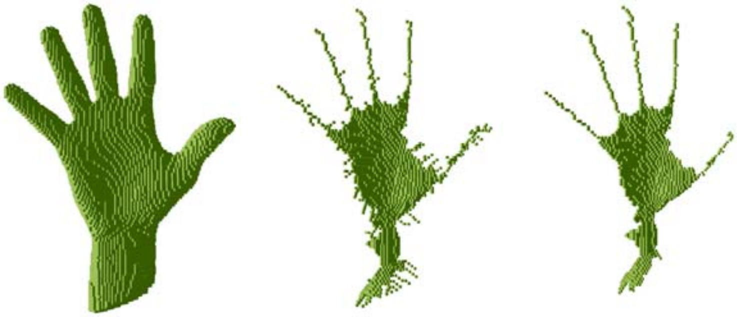

Fig. 3. An input 3D object, left, and the unit-wide set of patches and lines before pruning, middle, and after the 2 steps of pruning, right

Though PL has been reduced to unit-thickness, its structure may be still complex. We point out that a number of type2 voxels may exist, which do not constitute lines or do not border patches, perceived as corresponding to significant parts of the 3D input object. These type2 voxels can be understood as bordering peripheral patches or constituting peripheral lines jutting very little from the patch/line they are intersecting with. Peripheral patches and lines of PL having negligible perceptual meaning have to be removed, before computing the curvilinear skeleton. Otherwise, the structure of the curvilinear skeleton would deviate from the expected structure, and the skeleton would not provide a faithful enough representation of the object.

To get rid of peripheral patches jutting one voxel only, we remove all type2 voxels, provided that they are not necessary for topological reasons and are not tips of peripheral lines. Then, final thinning is applied again, since patches that were at most 4-voxel wide before removal of the type2 voxels, if any, might have been transformed into nearly-thin lines that have to be reduced to unit-thickness.

To remove non-significant peripheral lines, we resort to a pruning process, see e.g., [24,25]. A line of PL should be interpreted as significant if it corresponds to a region of the object that is perceived as important for shape analysis and recognition. In our opinion, since a strong relation exists between the centers of maximal balls along a line and size and shape of the object's region mapped into that line, the $\text{CMB1}_{\text{F}}\text{s}$ present on peripheral lines play a key role in deciding about pruning. Each peripheral line is traced starting from its end point, i.e., the voxel with only one object neighbor, until a branch point, i.e., a voxel with more than two object neighbors, is met. During tracing, the length L of the line, expressed in terms of the number of voxels constituting it, as well as the number N of $\text{CMB1}_{\text{F}}\text{s}$ along the line are computed. We use two pruning steps, performed with slightly different strategies. The first step is aimed at removing lines for which N is at most equal to a small threshold θ_1, to be fixed by the user (θ_1 has been set to 4 in this paper). As a result, lines with $L \leq \theta_1$, or lines for which, independently of their length L, is $N \leq \theta_1$ are removed. In both cases, these peripheral lines are interpreted as mapped into non-perceptually significant regions, due to the small number of $CMB1_F$s that they include. In particular, removal of peripheral lines with $L \leq \theta_1$ is crucial to get rid of short lines, whose presence would inhibit removal of lines that were non-peripheral in the initial PL, but have still to be interpreted as mapped into non-perceptually significant regions. During the second pruning step, peripheral lines, possibly originating from voxels that have modified their status into that of end points due to the first pruning step, are removed provided that it is $N/L \leq \theta_2$, where θ_2 is a threshold to be fixed depending on problem domain. We have experimentally found that $\theta_2=0.25$ is adequate as a default threshold. In Fig. 3 right, the pruned set PL resulting at the end of the first phase of the skeletonization process is shown.

2.2 Second Phase

The voxels of PL are classified during seven successive inspections of PL as follows:

Any voxel with at most two object neighbors that are not adjacent to each other is classified as *curve voxel.*

Any not yet classified voxel with a neighboring curve voxel is classified as *branching voxel.*

Any not yet classified voxel whose neighboring object voxels are all branching voxels is classified as *branching voxel.*

Any not yet classified voxel p with $c^*_p \neq 1$ is classified as *internal voxel.*

Any internal voxel p with more than two 6-connected components of background voxels in $N(p)$ face- or edge-adjacent to p, or being any of the eight object voxels in a 2×2×2 configuration is re-classified *as junction voxel.*

Any internal voxel having a face- or an edge-neighbor classified as junction voxel, is re-classified as *extended junction voxel.*

Any remaining not yet classified voxel is classified as *edge voxel.*

The classification allows us also to distinguish patches into peripheral patches (those at least partially delimited by edge voxels) from internal patches (those completely delimited by junction voxels).

The distance transform DT_{PL} of the set of voxels classified as internal in PL is computed in two steps by using the <3,4,5> distance function. During the first step, all edge voxels are taken as reference set, all internal voxels constitute the set of voxels to be assigned a distance value, while all background, curve, branching, junction and extended junction voxels act as barriers to the propagation of distance information. Actually, extended junction voxels, whose role is to better separate intersecting patches, are permitted to receive distance information, since after all they are voxels internal in a patch. However, they do not propagate distance information, to prevent interaction between intersecting patches for which voxels in a patch are neighbors of voxels in another patch. During the first step, only voxels internal in peripheral patches are assigned their distance from the edge voxels delimiting the patches. In fact, if internal patches exist in PL, no distance information may reach their internal voxels during the first step, due to the barriers of junction and extended junction voxels delimiting the internal patches. Thus, the second step is accomplished. Extended junction voxels are re-classified as internal voxels; junction voxels delimiting the internal patches are re-classified as *delimiting junction voxels* and are assigned a sufficiently large value (e.g., the maximum distance value assigned during the first step plus 5). All delimiting junction voxels are taken as reference set, all internal voxels that have not yet an assigned distance value constitute the set of voxels that will receive distance information, while all other voxels in the image act as barriers to the propagation of distance information.

In DT_{PL} the centers of maximal balls, CMB_{PL}, are detected with the same criterion used to detect the CMB_F s in DT_F.

Analogously to the first phase of skeletonization, only the $CMB1_{PL}$s, i.e., the CMB_{PL}s having no neighbors or at most vertex-neighbors with a distance value characterizing a more internal layer, should be taken as anchor points. Actually, we filter out the $CMB1_{PL}$s such that the homologous voxels in DT_F are not $CMB1_F$. In this way, the anchor points are by all means symmetrically placed within the original 3D object. In turn, differently from the first phase, the selected $CMB1_{PL}$s are not the only anchor points. In fact, also the voxels of PL classified as curve, branching or junction voxels are all considered as anchor points, as these voxels are placed in key positions as regards shape information.

The voxels in DT_{PL} are examined in increasing distance order, starting with those with the minimum distance value and, if they are not anchor points, topology preserving removal is sequentially accomplished. Only simple voxels are removed.

Once all distance values have been taken into account, the nearly-thin curvilinear skeleton CS is obtained.

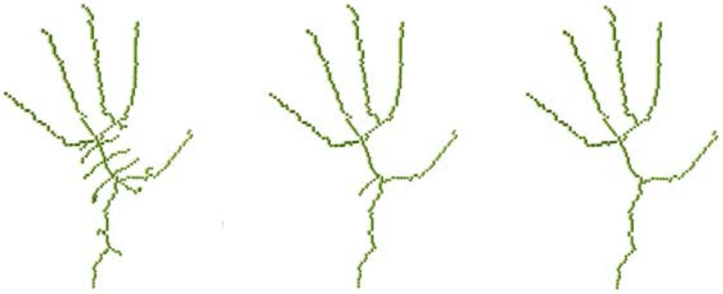

Fig. 4. The unit-wide curvilinear skeleton CS before pruning, left, and after the first step, middle, and the second step of pruning, right

The set CS is reduced to the unit-wide curvilinear skeleton by means of final thinning. Only voxels placed in 2-voxel thick configurations are possibly removed. Directional masks to detect thickening in face- or edge-direction are used as in the first phase. Any voxel *p* found to be placed in a thick configuration, according to the currently used directional mask, is sequentially removed provided that it is a simple voxel and has more than one neighboring object voxel. The result for the running example is shown in Fig. 4 left.

The set CS may still include some peripheral branches, which do not correspond to relevant parts of the 3D input object. Their removal is obtained by means of pruning, analogously to what has been done in the first phase. The only difference is that during the second pruning step the number N is now obtained by counting the selected CMB1_{PL}s as well as the remaining anchor points, i.e., the curve, branching and junction voxels along the peripheral branches. The values for the two thresholds θ_1 and θ_2 are here equal to those used in the first phase. In Fig. 4 middle and right, the curvilinear skeletons, as resulting after the first and the second pruning step, are shown.

3 Experimental Results

We have applied our skeletonization algorithm to a large collection of images taken from publicly available shape repositories [26,27]. A few examples of images are shown in Fig. 5 together with their curvilinear skeletons.

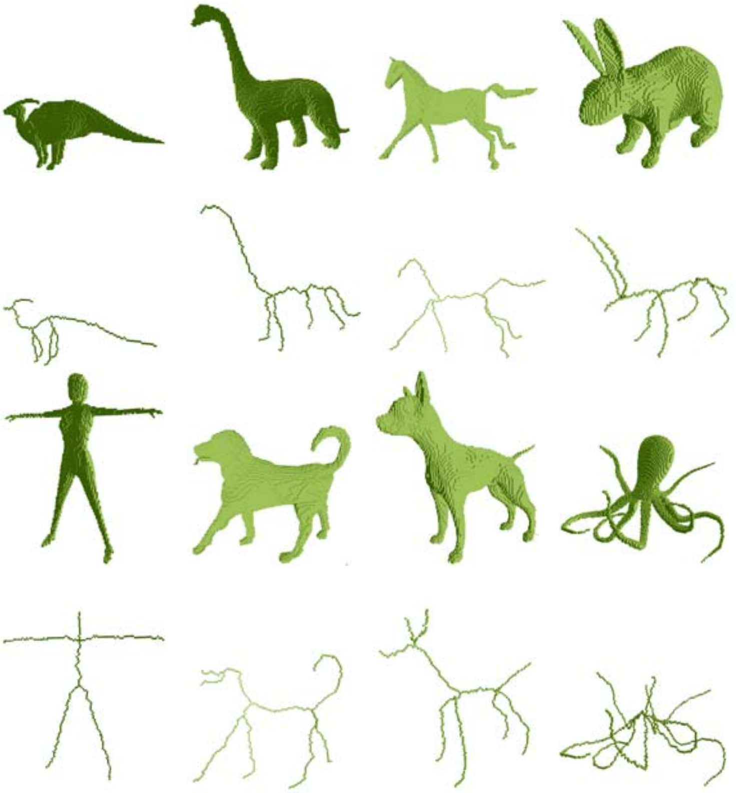

Fig. 5. Some 3D objects and their <3,4,5> curvilinear skeletons

Any skeletonization method should be evaluated in terms of the properties expected for the skeleton. The curvilinear skeleton is expected: 1) to be topologically equivalent to the object, 2) to be symmetrically placed within the object, and 3) to reflect the object's shape, by the presence of branches that can be seen as corresponding to parts of the object perceived as significant.

The first two properties are by all means satisfied by the <3,4,5> curvilinear skeleton. In fact, our voxel removal process has been accomplished (in both phases of skeletonization) sequentially, voxel after voxel, by using the notion of simple voxel. In turn, skeleton centrality is guaranteed by the criteria used in the two phases to select the anchor points, which are all symmetrically placed within the 3D object.

As for the third property, by looking at the examples in Fig. 5 we observe that the curvilinear skeletons have a number of branches corresponding to the main parts perceived as composing the objects. We remind that differently from the 2D case, the curvilinear skeleton does not generally enjoy the reversibility property. This means that by applying the reverse distance transformation to each single branch of the curvilinear skeleton we cannot completely recover the part of the 3D object mapped into that branch. However, the reverse distance transformation, computed by using for the voxels of the curvilinear skeleton the distance values pertaining to their homologous voxels in DT_F, can still be used to identify the main parts constituting the object, even if only in a rough manner.

As an example, the objects reconstructed from the curvilinear skeletons in Fig. 5 are given in Fig. 6. We note that the reconstructed objects may be regarded as sketched versions of the corresponding input objects. Thus, we can argue that the <3,4,5> curvilinear skeleton enjoys also the third property.

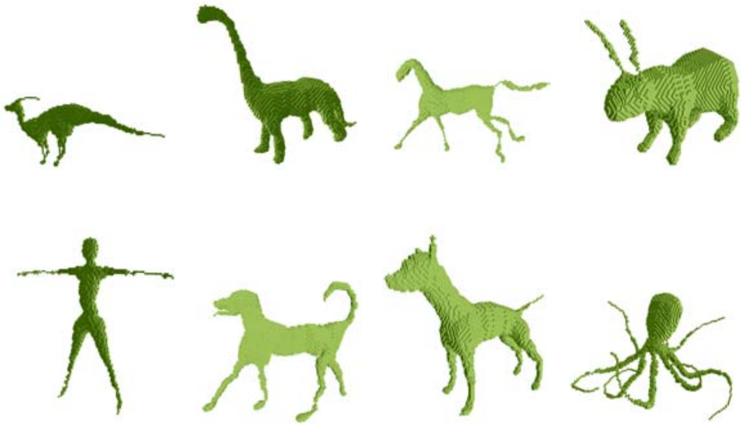

Fig. 6. Objects reconstructed form the curvilinear skeletons in Fig. 5

4 Concluding Remarks

In this paper, a method to compute the <3,4,5> curvilinear skeleton of 3D objects rid of cavities has been presented. Skeletonization is accomplished in two phases. The first phase is aimed at obtaining a subset of the 3D object consisting of 2D and 1D manifolds (patches and lines). The second phase is aimed at obtaining the curvilinear skeleton, exclusively consisting of 1D manifolds.

Both phases of the algorithm take advantage of the <3,4,5> distance transform. In the first phase, the distance transform DT_F of the object F is used to detect the anchor points (a suitable subset of the set of centers of maximal balls) and to guide iterated voxel removal (performed sequentially in increasing distance value order, and based on the notion of simple voxel). In the second phase, voxel classification of the set PL, resulting at the end of the first phase, is done to correctly identify the voxels constituting the reference set and the barriers. The distance transform DT_{PL} is computed and anchor points are detected as a subset of the set of center of the maximal balls as well as voxels classified as curve, branching and junction voxels. The distance transform DT_{PL} is also used to guide iterated voxel removal (performed sequentially in increasing distance value order, and based on the notion of simple voxel). Final thinning and pruning are also included in both phases of skeletonization.

The use of the <3,4,5> distance, which is a good approximation of the Euclidean distance, coupled with the pruning steps, makes the skeletonization algorithm rather stable under object rotation. Moreover, the use of the distance transform is advantageous from a computational point of view, since the voxels constituting the border of the object at any iteration of skeletonization are directly accessed by taking into account the distance values assigned to them.

References

1. Siddiqi, K., Pizer, S.M. (eds.): Medial Representations. Springer, Heidelberg (2008)
2. De Floriani, L., Spagnuolo, M. (eds.): Shape Analysis and Structuring. Springer, Heidelberg (2008)
3. Arcelli, C., Sanniti di Baja, G.: A width-independent fast thinning algorithm. IEEE Trans. on PAMI 7, 463–474 (1985)
4. Klein, F.: Euclidean skeletons. In: Proc. 5th Scandinavian Conf. Image Anal., pp. 443–450 (1987)
5. Arcelli, C., Sanniti di Baja, G.: A one-pass two-operations process to detect the skeletal pixels on the 4-distance transform. IEEE Trans. PAMI 11, 411–414 (1989)
6. Xia, Y.: Skeletonization via the realization of the fire front's propagation and extinction in digital binary shapes. IEEE Trans. PAMI 11(10), 1076–1086 (1989)
7. Arcelli, C., Sanniti di Baja, G.: Euclidean skeleton via center-of-maximal-disc extraction. Image and Vision Computing 11, 163–173 (1993)
8. Kimmel, R., Shaked, D., Kiryati, N.: Skeletonization via distance maps and level sets. Computer Vision and Image Understanding 62(3), 382–391 (1995)
9. Sanniti di Baja, G., Thiel, E.: Skeletonization algorithm running on path-based distance maps. Image and Vision Computing 14, 47–57 (1996)
10. Pudney, C.: Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images. Computer Vision and Image Understanding 72(3), 404–413 (1998)
11. Zhou, Y., Kaufman, A., Toga, A.W.: Three-dimensional skeleton and centerline generation based on an approximate minimum distance field. The Visual Computer 14(7), 303–314 (1998)
12. Borgefors, G., Nystrom, I., Sanniti di Baja, G.: Computing skeletons in three dimensions. Pattern Recognition 32(7), 1225–1236 (1999)

13. Sanniti di Baja, G., Svensson, S.: Surface skeletons detected on the d6 distance transform. In: Amin, A., Pudil, P., Ferri, F., Iñesta, J.M. (eds.) SPR 2000 and SSPR 2000. LNCS, vol. 1876, pp. 387–396. Springer, Heidelberg (2000)
14. Blum, H.: Biological shape and visual science. Journal of Theoretical Biology 38, 205–287 (1973)
15. Leymarie, F., Levine, M.D.: Simulating the grassfire transform using an active contour model. IEEE Trans. PAMI 14(1), 56–75 (1992)
16. Kimia, B.B., Tannenbaum, A., Zucker, S.W.: Shape, shocks, and deformations I: the components of two-dimensional shape and the reaction-diffusion space. International Journal of Computer Vision 15, 189–224 (1995)
17. Siddiqi, K., Bouix, S., Tannenbaum, A., Zucker, S.W.: Hamilton-Jacobi skeletons. International Journal of Computer Vision 48(3), 215–231 (2002)
18. Dimitrov, P., Damon, J.N., Siddiqi, K.: Flux invariants for shape. In: Proc. IEEE Conf. CVPR 2003, Madison, WI, vol. 1, pp. 835–841 (2003)
19. Giblin, P.J., Kimia, B.B.: A formal classification of 3D medial axis points and their local geometry. IEEE Trans. PAMI 26(2), 238–251 (2004)
20. Saha, P.K., Chaudhuri, B.B.: Detection of 3D simple points for topology preserving transformations with application to thinning. IEEE Trans. PAMI 16(10), 1028–1032 (1994)
21. Bertrand, G., Malandain, G.: A new characterization of three-dimensional simple points. Pattern Recognition Letters 15(2), 169–175 (1994)
22. Borgefors, G.: Digital distance transforms in 2D, 3D, and 4D. In: Chen, C.H., Wang, P.P.S. (eds.) Handbook of Pattern Recognition and Computer Vision, pp. 157–176. World Scientific, Singapore (2005)
23. Svensson, S., Sanniti di Baja, G.: Using distance transforms to decompose 3D discrete objects. Image and Vision Computing 20, 529–540 (2002)
24. Shaked, D., Bruckstein, A.M.: Pruning medial axes. Computer Vision and Image Understanding 69(2), 156–169 (1998)
25. Svensson, S., Sanniti di Baja, G.: Simplifying curve skeletons in volume images. Computer Vision and Image Understanding 90(3), 242–257 (2003)
26. AIM@SHAPE Shape Repository, `http://shapes.aimatshape.net/viewmodels.php`
27. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The Princeton Shape Benchmark. In: Shape Modeling International, Genova, Italy (June 2004)