Centerline Extraction from 3D Airway Trees Using Anchored Shrinking

Kálmán Palágyi(✉) and Gábor Németh

Department of Image Processing and Computer Graphics,
University of Szeged, Szeged, Hungary
{palagyi,gnemeth}@inf.u-szeged.hu

Abstract. Centerline is a frequently applied 1D representation of 3D tubular and tree-like objects. This paper proposes a new curve skeletonization algorithm, which is computationally efficient, guarantees 1-point wide centerlines, and does not generate 'spurious' branches. The reported method is specifically targeting segmented intrathoracic airway trees but it is applicable to many other tasks. Our algorithm is based on iterative shrinking combined with branch-end detection and preservation.

Keywords: Skeletonization · 3D Centerline · Digital topology · Shrinking · Medical image analysis

1 Introduction

Skeletonization provides region-based shape descriptors, which represent the topology and the general shape of objects [16]. 3D skeleton-like shape features (i.e., medial surface, centerlines, and topological kernel) play important role in various applications in image processing, pattern recognition, and visualization [14].

Surface skeletonization methods produce medial surfaces (i.e., union of 1D and 2D structures), curve skeletonization algorithms are used to extract centerlines (i.e., descriptors containing only 1D structures), and kernel skeletonization provides topological kernels (i.e., minimal sets of points that are topologically equivalent [7] to the original objects). Medial surfaces are usually extracted from general shapes, while tubular and tree-like objects can be represented by their centerlines, and topological kernels are useful in topological description [10,14,17,18].

Thinning is a frequently used approach to produce medial surfaces and centerlines in a topology-preserving way: the outmost layer of an object is deleted, and the entire process is repeated until stability is reached [7,14,17]. Topological kernels are generally produced by (reductive) shrinking [5]. Shrinking is similar to thinning: it is also an iterative object reduction, but geometric constraints are not taken into consideration. Parallel thinning and shrinking algorithms can

© Springer Nature Switzerland AG 2019
G. Bebis et al. (Eds.): ISVC 2019, LNCS 11845, pp. 419–430, 2019.
https://doi.org/10.1007/978-3-030-33723-0_34

delete a set of object points simultaneously, while sequential algorithms scan the picture, and focus on the actually visited single point for possible deletion [5].

Tubular and tree-like structures (e.g., arterial and venous systems, intrathoracic airways, and gastrointestinal tract) are frequently found in living organisms, thus centerlines (as 1D structures) can serve as viewpoint trajectory for navigation purposes in virtual angioscopy, bronchoscopy, or colonoscopy, and help us to generate formal structures for the forthcoming analysis and measurements [9,14,19,20].

In this study, our attention is focussed on the centerline extraction from 3D tree-like objects. Some recent reviews on curve skeletonization are available in [6,13,17,18]. Note that most of the existing centerline extraction methods are rather sensitive to coarse object boundaries, and may produce several spurious side branches. In order to overcome this problem, the false segments included by the produced centerlines are removed by a pruning process (i.e., a post-processing step) [15]. We should note that some existing algorithms are time consuming, and cannot produce 1-point wide centerlines for all possible objects.

This paper proposes a new curve skeletonization algorithm. It guarantees 1-point wide centerlines, preserves the topology, and it is computationally efficient. In addition, its centerlines are free from 'spurious' branches, hence no post-pruning is required. Our algorithm is based on shrinking combined with branch-end detection and preservation. The new method is compared with two existing curve skeletonization algorithms. The two older algorithms also have computationally efficient implementations and they can produce 1-point wide centerlines, too. It is illustrated that the new algorithm is as fast as the two existing algorithms under comparison, but it produces 'more reliable' results.

2 Basic Notions and Results

In this work, we apply the fundamental concepts of digital topology as reviewed by Kong and Rosenfeld [7].

Let p be a point in the 3D digital space $\mathbb{Z}^3$. Let us denote $N_j(p)$ (for $j = 6, 18, 26$) the set of points that are j-*adjacent* to point p and let $N_j^*(p) = N_j(p) \backslash \{p\}$, see Fig. 1.

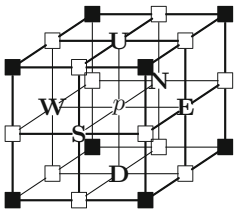


Fig. 1. Frequently used adjacencies in $\mathbb{Z}^3$. The set $N_6(p)$ contains point p and the six points marked **U**, **D**, **N**, **E**, **S**, and **W**. The set $N_{18}(p)$ contains $N_6(p)$ and the twelve points marked '□'. The set $N_{26}(p)$ contains $N_{18}(p)$ and the eight points marked '■'

The sequence of distinct points $\langle x_0, x_1, \ldots, x_n \rangle$ is called a *j-path* (for $j = 6, 18, 26$) of length n from point x_0 to point x_n in a non-empty set of points X if each point of the sequence is in X and x_i is j-adjacent to x_{i-1} for each $i = 1, \ldots, n$. (Note that a single point is a j-path of length 0.) Two points are said to be *j-connected* in the set X if there is a j-path in X between them. A set of points X is *j-connected* in the set of points $Y \supseteq X$ if any two points in X are j-connected in Y. A *j-component* of a set of points X is a maximal (with respect to inclusion) j-connected subset of X.

A $(26, 6)$ 3D binary digital picture on is a quadruple $(\mathbb{Z}^3, 26, 6, B)$, where $B \subseteq \mathbb{Z}^3$ is the set of black points (consequently, $\mathbb{Z}^3 \setminus B$ is the set of white points), 26-adjacency and 6-adjacency are used for B and $\mathbb{Z}^3 \setminus B$, respectively. For practical purposes, we assume that all pictures are *finite* (i.e. they contain finitely many black points).

A *black component* or *object* is a 26-component of B, while a *white component* is a 6-component of $\mathbb{Z}^3 \setminus B$. In a finite picture there is a unique infinite white component, which is called the *background*. A finite white component is said to be a *cavity*.

A black point is called a *border point* in a $(26, 6)$ picture if it is 6-adjacent to at least one white point. A border point is said to be a **U**-*border point* if the point marked **U** in Fig. 1 is white. We can define **D**-, **N**-, **E**-, **S**-, and **W**-border points in the same way. A black point is called an *interior point* if it is not a border point.

A *reduction* transforms a binary picture only by changing some black points to white ones (which is referred to as the deletion of black points). A reduction is *not* topology-preserving [7] if any object in the input picture is split (into several ones) or is completely deleted, any cavity in the input picture is merged with the background or another cavity, or a cavity is created where there was none in the input picture. There is an additional concept called *tunnel* (which donuts have) in 3D pictures [7]. Topology preservation implies that eliminating or creating any tunnel is not allowed.

A black point is *simple* if and only if its deletion is a topology-preserving reduction [7]. The following theorem states a characterization of simple points in $(26, 6)$ pictures:

Theorem 1. [8,12] *A black point p is simple in picture $(\mathbb{Z}^3, 26, 6, B)$ if and only if all of the following conditions hold:*

1. *The set $N_{26}^*(p) \cap B$ contains exactly one 26–component.*
2. *The set $N_6(p) \setminus B$ is not empty (i.e., p is a border point).*
3. *Any two points in $N_6(p) \setminus B$ are 6–connected in the set $N_{18}(p) \setminus B$.*

Based on Theorem 1, the simplicity of a point p can be decided by examining the set $N_{26}^*(p)$ (i.e., it is a local property).

3D curve-thinning algorithms generally preserve curve-end points:

Definition 1. *A (simple) black point p in picture $(\mathbb{Z}^3, 26, 6, B)$ is a* curve-end point of type *if the set $N_{26}^*(p) \cap B$ contains exactly one point (i.e., p is 26-adjacent to exactly one further black point).*

Bertrand and Couprie proposed an alternative approach for curve-thinning by accumulating some curve interior points that are called isthmuses [2].

Definition 2. *A border point p in a picture $(\mathbb{Z}^3, 26, 6, B)$ is a* curve-isthmus *if the set $N^*_{26}(p) \cap B$ contains more than one 26–component.*

All curve-isthmuses are not simple points since Condition 1 of Theorem 1 is violated. Note that the characterization of curve-isthmuses examines the set $N^*_{26}(p)$ for a point p in question.

Figure 2 presents examples of simple, curve-end, and curve-isthmus points.

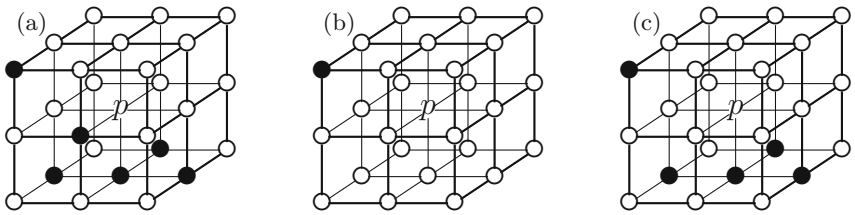


Fig. 2. Configuration in which the central black point p is a simple point (a), a curve-end point (b), and a curve-isthmus point (c). Black and white points are marked '●' and '○', respectively

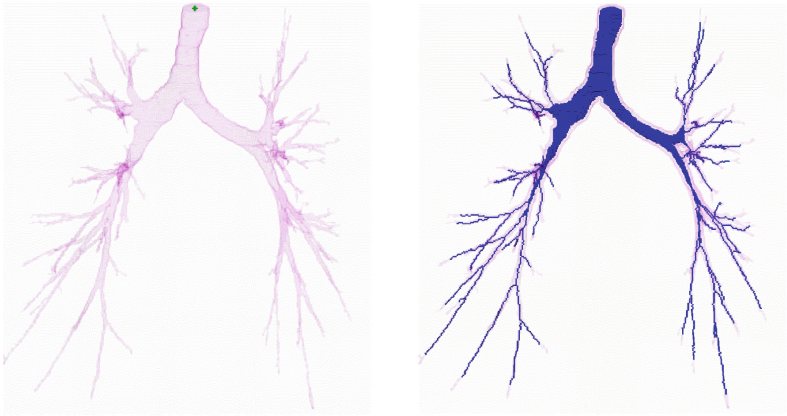
3 Curve Skeletonization Based on Shrinking

In this section the curve skeletonization algorithm **3D-CS-APS** is presented. The reported method is specifically targeting segmented intrathoracic airway trees but it is applicable to many other tasks. The input of our algorithm is a 3D $(26, 6)$ picture representing a segmented volumetric tree object. The method consists of the following four steps:

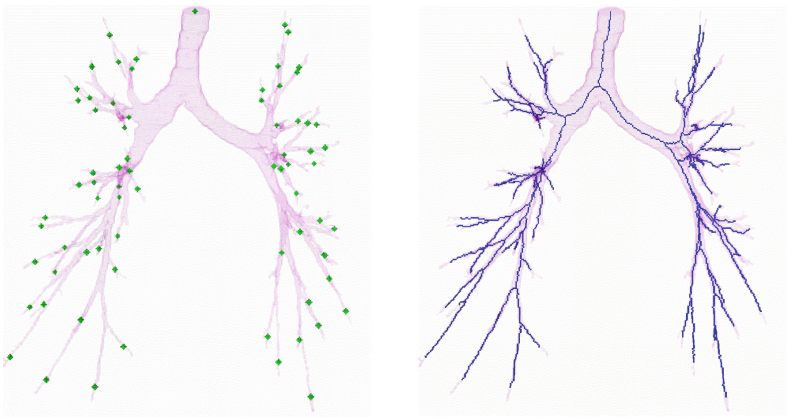
1. identification of the tree root as an anchor point (see Fig. 3a),
2. the first k iterations of anchor-preserving shrinking (see Fig. 3b),
3. detection of curve-end points as further anchor points (see Fig. 3c), and
4. anchor-preserving shrinking until stability is reached (see Fig. 3d).

These steps are now described in more detail.

Step 1: Root Detection
The root detection is not a critical phase of the process. Since we deal with intrathoracic airway trees segmented from volumetric CT (or MR) image data, a priori knowledge of the data set can be used to identify the tree root. In [9], the center of the topmost nonzero 2D slice in direction z (detected by 2D shrinking)



(a) identification of the root (b) $k = 2$ iterations of shrinking



(c) detection of curve-end points (d) produced centerline

Fig. 3. The four steps of algorithm **3D-CS-APS** for a human airway tree. The detected anchor points (a,c) and the resulted structures (b,d) are superimposed on the input tree

defines the root of the tree. (Note that the identified root point belongs to the trachea.) In other applications, different root identification approaches may be needed.

The detected root point (see Fig. 3a) acts as an *anchor point* in the remaining steps of the process (i.e., it cannot be deleted further on).

Step 2: Anchor Preserving Interrupted Shrinking
We make use of the fact that the trachea of a human airway tree is the thickest branch, and the terminating branches are the thinnest ones. Hence a given number of shrinking iterations produce 1-point wide terminating branches that end with curve-end points, and no further curve-end points are generated in the 'thicker' branches (see Fig. 3b).

Many researchers proposed 3D parallel shrinking algorithms [1,10]. Here we present a new 6-subiteration sequential anchor-preserving shrinking algorithm capable of producing 1-point wide structures. Hence, centerlines produced by algorithm **3D-CS-APS** contain only non-simple points and (simple) curve-end points.

One iteration step of our anchor-preserving shrinking is outlined by Algorithm 1.

Algorithm 1: One Iteration Step of the Anchor-Preserving Shrinking

Input: picture $(\mathbb{Z}^3, 26, 6, X)$ and set of anchor points $Anchors \subseteq X$
Output: picture $(\mathbb{Z}^3, 26, 6, Y)$
$Y \leftarrow X$
foreach direction $d \in \{$**U**,**N**,**E**,**S**,**W**,**D**$\}$ **do**
  `// Subiteration according to the deletion direction` d
  `// Collecting potentially deletable points`
  $Z \leftarrow \emptyset$
  foreach point $p \in (Y \setminus Anchors)$ **do**
    if p is a d-border and simple point in $(\mathbb{Z}^3, 26, 6, Y)$ **then**
      $Z \leftarrow Z \cup \{p\}$
  `// Deletion`
  foreach point $p \in Z$ **do**
    if p is a simple point in $(\mathbb{Z}^3, 26, 6, Y)$ **then**
      $Y \leftarrow Y \setminus \{p\}$

Algorithm 1 (i.e., one iteration step of our anchor-preserving shrinking algorithm) is decomposed into six successive subiterations according to the six main directions in 3D, and each subiteration consists of two phases. First, black points that are not anchor points are traversed. If the given point is a border point of the actual type and simple point (in the input picture), it is marked as a potentially deletable point. In the second phase, a marked point is deleted if it remains simple after the deletion of the previously visited marked points.

Depending on the 'thickness' of the terminating branches of the segmented tree, we apply k iterations of shrinking (and the iterative peeling is interrupted). In this step, the set of anchor points is singleton in each iteration, it contains only the root point (detected in the first step of the process).

Step 3: Detection of Curve-End Points
This step is fairly straightforward: all curve-end points (see Definition 1) in the structure produced by the second step of the process are to be identified (see Fig. 3c). These points and the root point form the set of anchor points for the last step.

Step 4: Anchor Preserving Final Shrinking
The proposed process is completed by anchor preserving shrinking until stability is reached. This step forms 1-point wide centerlines of airway trees by connecting the root point and the ends of terminating branches (see Fig. 3d).

Since the proposed sequential shrinking algorithm may delete just one simple point at a time, the entire process preserves the topology.

4 Computationally Efficient Implementation

One may think that the proposed curve skeletonization algorithm (see Sect. 3) is time-consuming, and it is rather difficult to implement it. That is why Algorithm 2 presents a computationally efficient implementation of the proposed algorithm **3D-CS-APS**. Note that similar implementation schemes were proposed by Palágyi et al. [9,10] for arbitrary sequential and parallel thinning algorithms.

The input of Algorithm 2 is array A which stores the $(26, 6)$ picture with the tree-like object to be represented. In input array A, the value '1' corresponds to black points and the value '0' is assigned to white ones. According to the proposed scheme, the input and the output pictures can be stored in the same array, hence array A will contain the produced centerline.

We use two lists to speed up the process: *border_list* stores the border points in the current picture (hence the repeated scans of the entire array A are avoided), and *potentially_deletable_list* is to collect all potentially deletable points in the current subiteration of the anchor preserving shrinking. (Note that *potentially_deletable_list* is a sublist of *border_list*.) In order to avoid storing more than one copy of a border point in *border_list*, and examining anchor points for possible deletion, array A represents a four-color picture:

- the value of '0' corresponds to white points,
- the value of '1' is assigned to (black) interior points,
- the value of '2' corresponds to (black) border points in the actual picture (i.e., elements of *border_list*), and
- the value of '3' is assigned to the detected (black) anchor points.

First, the root point (i.e., the initial anchor point) is identified by the function 'ROOT_DETECTION'. Then the input picture is scanned, and all the border points in it are inserted into the list *border_list*. We should note that it is the only time-consuming scan in the entire process.

The next step is the first k iterations of anchor-preserving shrinking. We use a pre–calculated look-up-table to encode simple points. Simple points in $(26, 6)$ pictures can be locally characterized; this property for a point p can be decided by examining the set $N^*_{26}(p)$ that contains 26 points (see Theorem 1). Hence the pre-calculated look-up-table has 2^{26} entries of 1 bit in size. It is clear that our look-up-table requires just 8 megabytes of storage space in memory.

Algorithm 2: Efficient Implementation of Algorithm **3D-CS-APS**

```
Input: array A storing the segmented tree and number of iterations k
Output: array A containing the picture with the produced centerline
// Step 1 - identification of the tree root
(x, y, z) ← ROOT_DETECTION (A);  A[x, y, z] ← 3;
// Collect border points
border_list ← <empty list>;
foreach p = (x, y, z) in array A do
    if A[x, y, z] = 1 and p is a border point then
        border_list ← border_list + <p>;  A[x, y, z] ← 2;
// Step 2 - the first k iterations of anchor-preserving shrinking
for i ← 1 to k do
    foreach direction d ∈ {U,N,E,S,W,D} do
        potentially_deletable_list ← <empty list>;
        foreach point p in border_list do
            if p is a d-border and simple point then
                potentially_deletable_list ← potentially_deletable_list + <p>;
        foreach point p = (x, y, z) in potentially_deletable_list do
            if p is a simple point then
                A[x, y, z] ← 0;  border_list ← border_list − <p>;
                foreach point q = (x', y', z') that is 6-adjacent to p do
                    if A[x', y', z'] = 1 then
                        A[x', y', z'] ← 2;  border_list ← border_list + <q>;
// Step 3 - detection of curve-end points
foreach point p = (x, y, z) in border_list do
    if p is a curve-end point then
        A[x, y, z] ← 3;  border_list = border_list − <p>;
// Step 4 - anchor-preserving shrinking
repeat
    number_of_deleted_points = 0;
    foreach direction d ∈ {U,N,E,S,W,D} do
        potentially_deletable_list ← <empty list>;
        foreach point p in border_list do
            if p is a d-border and simple point then
                potentially_deletable_list ← potentially_deletable_list + <p>;
        foreach point p = (x, y, z) in potentially_deletable_list do
            if p is a simple point then
                A[x, y, z] ← 0;  border_list = border_list − <p>;
                number_of_deleted_points = number_of_deleted_points +1;
                foreach point q = (x', y', z') that is 6-adjacent to p do
                    if A[x', y', z'] = 1 then
                        A[x', y', z'] ← 2;  border_list ← border_list + <q>;
until number_of_deleted_points = 0;
```

Each subiteration of the anchor preserving shrinking is a 2-phase process: First, simple border points of the actual type that are added to the *potentially_deletable_list*. In the second phase, a point in the *potentially_deletable_list* is deleted if it remains simple after the deletion of the previously visited and deleted points. If a border point is deleted, all interior points that are 6-adjacent to it become border points. These brand new border points of the resulted picture are added to the *border_list*.

The third step of algorithm **3D-CS-APS** is fairly simple. Since all curve-end points are border points, only points in the *border_list* are to be examined. The detected curve-end points are removed from the *border_list* and the value of '3' is assigned to these new anchor points.

The last step of algorithm **3D-CS-APS** is the ultimate anchor preserving shrinking. The number of deleted points within an iteration step is stored in the variable *number_of_deleted_points*. The algorithm terminates when stability is reached (i.e., *number_of_deleted_points* = 0). Then all points having a nonzero value belong to the produced centerline.

5 Experiments

In experiments, the proposed algorithm **3D-CS-APS** is compared with two existing curve skeletonization algorithms:

- Palágyi et al. [9] suggested a 6-subiteration sequential curve thinning algorithm named **PTHS_2006**. It uses endpoint-rechecking: a curve-end point (see Definition 1) can be deleted if at least t points of its 6-neighbors have been deleted during the actual subiteration. According to the experiences, setting $t = 1$ is suggested for human airway trees.
- In [11], Palágyi proposed an isthmus-based 6-subiteration sequential curve thinning algorithm named **P_2014**. This algorithm accumulates and preserves curve-isthmus points (see Definition 2), while non-accumulated curve-end points are deleted.

We selected these two existing algorithms, since they also have computationally efficient implementations. In addition, the selected algorithms can also produce 1-point wide centerlines for all possible 3D objects.

The three curve skeletonization algorithms (i.e., **PTHS_2006**, **P_2014**, and **3D-CS-APS**) were tested on various segmented intrathoracic airway trees. Due to the lack of space, here we can present just two illustrative examples, see Figs. 4 and 5. We can state that the existing algorithms **PTHS_2006** and **P_2014** generate some unwanted side branches due to the coarse object boundary. Thanks to the anchor-preserving shrinking, centerlines produced by the proposed algorithm **3D-CS-APS** are free from 'spurious' branches. Hence, our method do not require post-pruning. We should note that algorithm **3D-CS-APS** slightly trims the terminating branches.

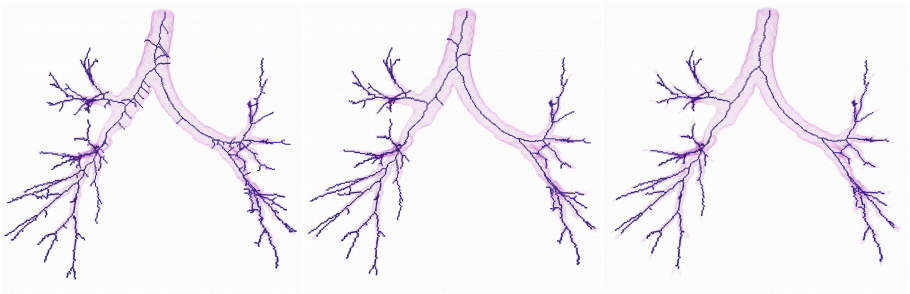


Fig. 4. Centerlines produced by the three curve skeletonization algorithms superimposed on a $512 \times 512 \times 509$ image of a segmented human airway tree (with 'noisy' boundary)

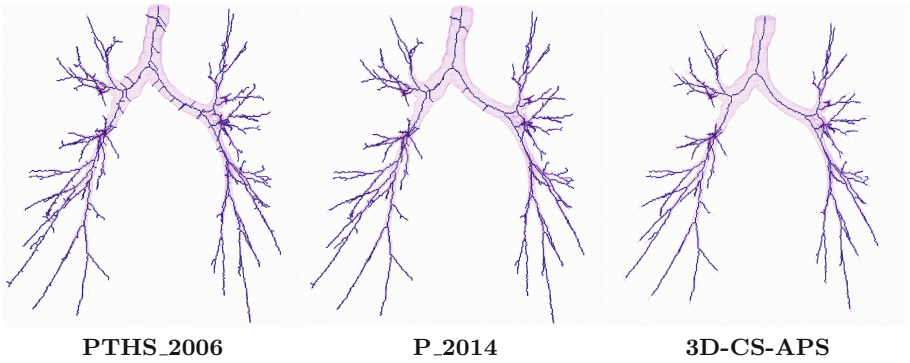


Fig. 5. Centerlines produced by the three curve skeletonization algorithms superimposed on a $512 \times 512 \times 490$ image of a segmented human airway tree (with 'noisy' boundary)

Table 1 illustrates the efficiency of the proposed algorithm **3D-CS-APS** on 12 human airway trees, and validates quantitatively its superiority over the two examined two methods. The examined algorithms were run on a usual PC under Linux (Fedora 27 - 64 bit), using a 3.30 GHz 4x Intel Core i5-2500 CPU. (Note, that just the centerline extraction process itself is considered here; reading the input volume and writing the output image are not taken into account.) We can state that the computational complexity of the selected three curve skeletonization algorithms are extremely low, they are much faster than the concurrent methods [3,4,6,19,20].

Table 1. Computation time (in sec.) by the three algorithms on 12 human airway trees (with 'noisy' boundaries). The image size ($512 \times 512 \times dim\_z$) and the number of branches (#branch) in the produced centerlines and are also given. Parameter settings: $t = 1$ for **PTHS_2006**, and $k = 1$ for **3D-CS-APS**. Note that the proposed algorithm **3D-CS-APS** does not produce visible unwanted side branches, and do not miss visually obvious branches.

Test object	Size (dim_z)	PTHS_2006		P_2014		3D-CS-APS	
		Time	#branch	Time	#branch	Time	#branch
Tree 1	436	0.31	120	0.30	93	0.29	75
Tree 2	537	0.36	111	0.35	90	0.35	74
Tree 3	557	0.34	37	0.34	32	0.33	27
Tree 4	490	0.30	44	0.30	38	0.29	31
Tree 5	490	0.33	135	0.32	116	0.32	93
Tree 6	459	0.29	57	0.29	49	0.28	42
Tree 7	387	0.24	26	0.23	22	0.23	19
Tree 8	349	0.22	45	0.22	31	0.21	23
Tree 9	389	0.25	53	0.24	37	0.24	32
Tree 10	509	0.34	83	0.33	72	0.32	61
Tree 11	509	0.40	124	0.38	100	0.39	82
Tree 12	556	0.34	63	0.33	47	0.32	41

6 Conclusions

In this paper, a new algorithm is proposed for producing centerlines from binary 3D tree-like objects. It is based on interrupted anchor-preserving shrinking, hence the produced centerlines do not contain 'spurious' branches. Our method is usable for tree-like objects in which each 'important' terminating branch ends with a curve-end points after a given number of shrinking iterations. The proposed algorithm is computationally efficient and preserves the topology. Quantitative experiments on intrathoracic airway trees have demonstrated that the new method outperforms two existing curve skeletonization algorithms.

Acknowledgments. This research was supported by the project "Integrated program for training new generation of scientists in the fields of computer science", no EFOP-3.6.3-VEKOP-16-2017-0002. The project has been supported by the European Union and co-funded by the European Social Fund.

References

1. Bertrand, G., Couprie, M.: Powerful parallel and symmetric 3D thinning schemes based on critical kernels. J. Math. Imaging Vis. **48**, 134–148 (2014)

2. Bertrand, G., Couprie, M.: Transformations topologiques discrètes. In: Coeurjolly, D., Montanvert, A., Chassery, J. (eds.) Géométrie discrète et images numériques, pp. 187–209. Hermès Science Publications (2007)
3. Bitter, I., Kaufman, A.E., Sato, M.: Penalized-distance volumetric skeleton algorithm. IEEE Trans. Vis. Comput. Graph. **7**, 195–206 (2001)
4. Bouix, S., Siddiqi, K., Tannenbaum, A.: Flux driven automatic centerline extraction. Med. Image Anal. **9**, 209–221 (2005)
5. Hall, R.W., Kong, T.Y., Rosenfeld, A.: Shrinking binary images. In: Kong, T.Y., Rosenfeld, A. (eds.) Topological Algorithms for Digital Image Processing, pp. 31–98. Elsevier Science B. V. (1996)
6. Jin, D., Chen, C., Hoffman, E.A., Saha, P.K.: Curve skeletonization using minimum-cost path. In: Saha, P.K., Borgefors, G., Sanniti di Baja, G. (eds.) Skeletonization: Theory, Methods and Applications, pp. 151–180. Academic Press (2017)
7. Kong, T.Y., Rosenfeld, A.: Digital topology: introduction and survey. Comput. Vis. Graph. Image Process. **48**, 357–393 (1989)
8. Malandain, G., Bertrand, G.: Fast characterization of 3D simple points. In: Proceedings of the 11th IEEE International Conference on Pattern Recognition, ICPR 1992, pp. 232–235 (1992)
9. Palágyi, K., Tschirren, J., Hoffman, E.A., Sonka, M.: Quantitative analysis of pulmonary airway tree structures. Comput. Biol. Med. **36**, 974–996 (2006)
10. Palágyi, K., Németh, G., Kardos, P.: Topology preserving parallel 3D thinning algorithms. In: Brimkov, V.E., Barneva, R.P. (eds.) Digital Geometry Algorithms. Theoretical Foundations and Applications to Computational Imaging, pp. 165–188. Springer, Dordrecht (2012). https://doi.org/10.1007/978-94-007-4174-4_6
11. Palágyi, K.: A sequential 3D curve-thinning algorithm based on isthmuses. In: Bebis, G., et al. (eds.) ISVC 2014. LNCS, vol. 8888, pp. 406–415. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-14364-4_39
12. Saha, P.K., Chaudhury, B.B.: Detection of 3-D simple points for topology preserving transformations with application to thinning. IEEE Trans. Pattern Anal. Mach. Intell. **16**, 1028–1032 (1994)
13. Saha, P.K., Borgefors, G., Sanniti di Baja, G.: A survey on skeletonization algorithms and their applications. Pattern Recogn. Lett. **76**, 3–12 (2016)
14. Saha, P.K., Borgefors, G., Sanniti di Baja, G. (eds.): Skeletonization: Theory, Methods and Applications. Academic Press, Cambridge (2017)
15. Shaked, D., Bruckstein, A.: Pruning medial axes. Comput. Vis. Image Underst. **69**, 156–169 (1998)
16. Siddiqi, K., Pizer, S. (eds.): Medial Representations – Mathematics, Algorithms and Applications. Computational Imaging and Vision, vol. 37. Springer, Dordrecht (2008). https://doi.org/10.1007/978-1-4020-8658-8
17. Sobiecki, A., Jalba, A., Telea, A.: Comparison of curve and surface skeletonization methods for voxel shapes. Pattern Recogn. Lett. **47**, 147–156 (2014)
18. Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., Telea, A.: 3D skeletons: a state-of-the-art report. In: Proceedings of the Conference on European Association for Computer Graphics, EG 2016, pp. 573–597 (2016)
19. Wan, M., Liang, Z., Ke, Q., Hong, L., Bitter, I., Kaufman, A.: Automatic centerline extraction for virtual colonoscopy. IEEE Trans. Med. Imaging **21**, 1450–1460 (2002)
20. Wong, W.C.K., So, R.W.K., Chung, A.C.S.: Principal curves for lumen center extraction and flow channel width estimation in 3-D arterial networks: theory, algorithm, and validation. IEEE Trans. Image Process. **21**, 1847–1862 (2012)