

Quantitative analysis of three-dimensional tubular tree structures

Kalman Palagyi, Juerg Tschirren, Milan Sonka

Dept. of Electrical and Computer Engineering, The University of Iowa,
4016 Seamans Center, Iowa City, IA 52242-1595

ABSTRACT

Quantitative assessment of tree structures is very important for evaluation of airway or vascular tree morphology and its associated function. Our skeletonization and branch-point identification method provides a basis for tree quantification or tree matching, tree-branch diameter measurement in any orientation, and labeling individual branch segments. All main components of our method were specifically developed to deal with imaging artifacts typically present in volumetric medical image data. The proposed method has been tested in a computer phantom subjected to changes of its orientation as well as in a repeatedly CT-scanned rigid plastic phantom. In all cases, our method produced reliable and well positioned centerlines and branch-points.

Keywords: skeletonization, centerline extraction, CT, airway trees

1. INTRODUCTION

Tubular tree structures are common in human anatomy. Arterial, venous, or bronchial trees may serve as most frequent examples. Computed tomography (CT) or magnetic resonance (MR) imaging provides volumetric image data allowing identification of such tree structures. Frequently, the trees represented as contiguous sets of voxels must be quantitatively analyzed. The analysis may be substantially simplified if the voxel-level tree is represented in a formal tree structure consisting of a set of nodes and connecting arcs. To build such formal trees, the voxel-level tree object must be transformed into a set of interconnected single-voxel centerlines representing individual tree branches. Therefore, the aim of our work was to develop a robust method for identification of centerlines and bifurcation (trifurcation, etc.) points in segmented tubular tree structures acquired in vivo from humans and animals using volumetric CT or MR scanning.

2. METHODS

The input of the proposed method is a 3D binary image representing a segmented voxel-level tree object. The entire tree analysis process consists of the following main steps: topological correction, root detection, centerline extraction by thinning, pruning, branch-point identification, generating formal tree structures representing centerlines and branches, and branch labeling.

2.1. Topological Correction of the Segmented Tree

When applied to clinical volumetric images, segmentation algorithms may produce imperfect segmentation results in which the segmented objects contain internal cavities (i.e., connected “0” voxels surrounded by “1” voxels), holes (i.e., “0” voxels forming a tunnel such as a doughnut has), and bays (i.e., disturbances without a topological meaning). Some of them cause unwanted changes of the underlying topology, all of them disturb and consequently yield an incorrect set of centerlines and thus an incorrect formal representation. To overcome the effects of artifactual cavities, the “0” voxels connected to the frame of the volume are labeled by sequential forward and backward scanning (instead of the conventional object labeling) then all unlabeled “0” voxels are filled (i.e., changed to “1” voxels). The applied method is similar to the linear-time Chamfer distance mapping¹. As a result, all cavities are filled with no connectivity alteration.

Holes and bays were removed by applying morphological closing² (i.e., a dilation followed by an erosion with a suitable structuring element). Note that the closing is a double-edged sword; it is suitable for filling small gaps, holes, and cavities, but new holes may be created. This side-effect can be handled by the pruning process that follows the centerline extraction. Fig. 1 shows an example of closing.

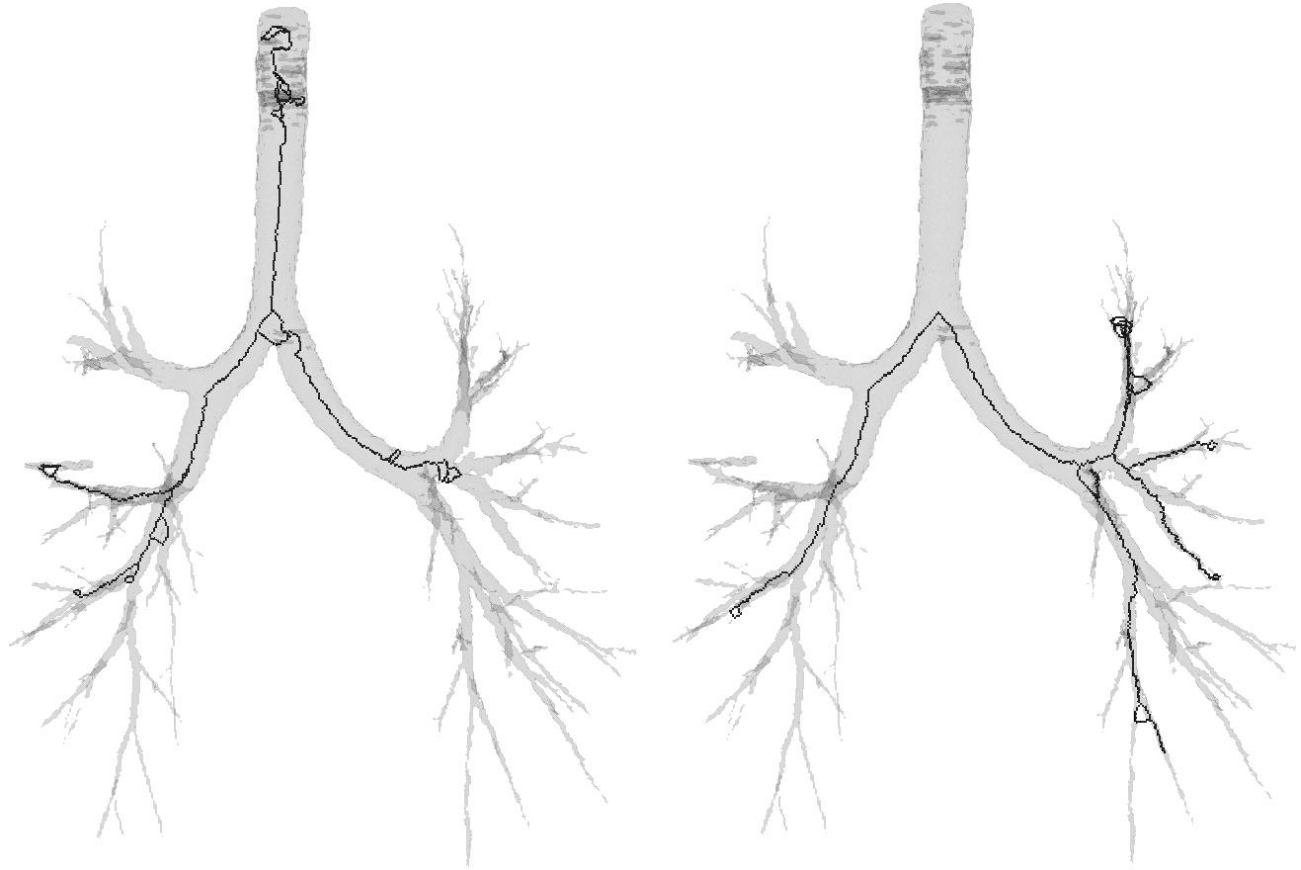


Figure 1. The segmented volume of a human airway tree (after cavity filling) and its topological kernel (i.e., the minimal structure being topologically equivalent to the original object) before morphological closing (left) and the tree and its topological kernel after closing (right). The topological kernel of a simply-connected object (i.e., object with no holes nor cavities) is a single voxel. Our sample tree contains some holes, therefore, the loops corresponding to them are connected in the topological kernel by 1-voxel wide line segments. The most disturbing holes and bays in lower generations were successfully filled, but some new holes were created in the higher generations. Such holes are later eliminated by the pruning process. The structuring element of the applied dilation and erosion corresponds to a $5 \times 5 \times 5$ voxel cube.

2.2. Root Detection

The center of the topmost nonzero 2D slice in direction z (detected by 2D shrinking) defines the root of the formal tree to be generated. In airway trees, the root point belongs to the trachea. The detected root voxel acts as an anchor point during the centerline extraction (i.e., it cannot be deleted by the forthcoming iterative peeling process).

The root detection is not a critical phase of the process. It can be identified interactively or automatically before segmentation³.

2.3. Centerline Extraction

One of the well-known approaches to centerline determination is to construct a 3D skeleton of the analyzed object. However, some of the properties of 3D skeletons in discrete grids are undesirable. Specifically, in the case of 3D

tubular objects, we do not need the exact skeleton, since skeleton generally contains surface patches. We need a skeletonization method that can suppress creation of such skeleton surface patches. As a solution, a 3D curve-thinning algorithm was developed that is preserving line-end points and can thus extract both geometrically and topologically correct centerlines. As part of this process, a novel method for endpoint re-checking was developed based on comparisons between the centerline configuration at some stage of thinning and the previous object configuration.

Thinning is a frequently used method for producing an approximation to the skeleton in a topology-preserving way⁴. Border points of a binary object that satisfy certain topological and geometric constraints are deleted in the iteration steps. In case of tubular 3D objects, thinning has a major advantage over other skeletonization methods since curve-thinning (i.e., iterative object reduction preserving line-end points) can produce one voxel wide centerlines directly⁵.

In order to outline our thinning scheme, let's first define (26, 6) images, border points (corresponding to the six main directions in 3D), line-end points, and simple points. A binary image is (26, 6) image if 26-connectivity (i.e., the reflexive and transitive closure of the 26-adjacency relation) is considered for "1" voxels forming the objects and 6-connectivity (i.e., the reflexive and transitive closure of the 6-adjacency) is considered for "0" voxels⁴. A "1" voxel in a (26, 6) image is called U-border point if its 6-neighbor in direction U ("up") is "0". We can define N-, E-, S-, W-, and D-border ("down") points in the same way. A "1" voxel is called line-end point if it has exactly one "1" 26-neighbor. A "1" voxel is called a simple point if its deletion does not alter the topology of the image⁴. It needs to be emphasized that simplicity in (26, 6) images is a local property that can be decided by investigating the 26-neighbors, the $3 \times 3 \times 3$ neighborhood of any given point⁶.

Our sequential thinning algorithm can be regarded as a modified version of the method proposed by Lee, Kashyap, and Chu⁷. It is sketched as follows:

```

repeat
  for each direction U, N, E, S, W, and D do
    mark all border points according to the actual direction that are
    simple points and not line-end points
    for each marked point  $p$  do
      if  $p$  is simple in the actual image then
        if  $p$  is not a line-end point then
          delete  $p$ 
        else if  $\#(\text{deleted 6-neighbors of } p) \geq t$  then
          delete  $p$ 
      endifor
    endifor
  endifor
until changes occur

```

One iteration step of the sequential object reduction process (i.e., the kernel of the **repeat** cycle) is decomposed into six successive sub-iterations according to the six main directions in 3D. Each sub-iteration consists of two phases; at first the border points of the actual type that are simple and not line-end points are marked as potential deletable points of the actual sub-iteration. This phase of the algorithm can be executed in parallel, but the forthcoming re-checking phase must be sequential. During the re-checking, a marked point is deleted if it remains simple and is not a line-end point after the deletion of some previously visited marked points. In addition, in the following special cases, simple points are deleted if they have become line-end point, too. The algorithm uses an extra parameter $t \in \{0, 1, 2, 3, 4, 5, 6\}$ and a marked (simple and line-end) point can be deleted if at least t points of its 6-neighbors have been deleted during the actual phase of the process. Note that in the case of $t = 6$, the novel endpoint re-checking has no effect (since a point is not simple if all of its 6-neighbors belong to the object). In that case, it produces the same result as the method proposed by Lee, Kashyap, and Chu⁷. According to our experience, setting $t = 1$ or $t = 2$ is suggested for human airway trees. See Fig. 2 for an example of the usefulness of the endpoint re-checking.

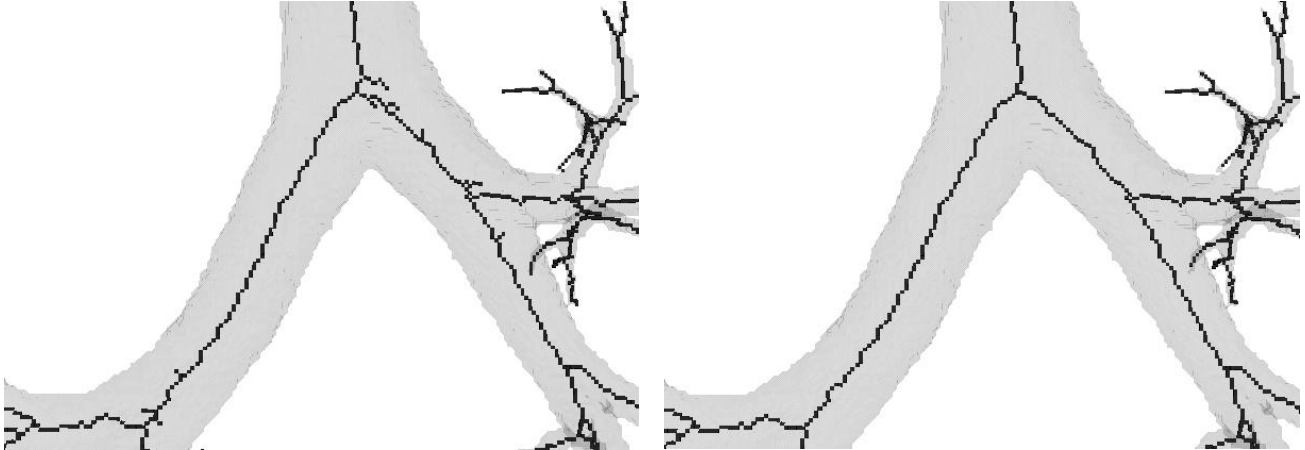


Figure 2. A part of a segmented tree and its raw centerline extracted by the proposed algorithm without endpoint re-checking ($t = 6$) (left). The result with endpoint re-checking ($t = 1$) (right). The centerline extracted by endpoint re-checking contains only 125 (true) branch-points (junctions) and 128 line-end points. There are 167 branch-points and 176 line-end points in the centerline generated without endpoint re-checking.

2.4. Pruning

Unfortunately, each skeletonization algorithm (including ours) is rather sensitive to coarse object boundaries. As a result, the produced (approximation to the) skeleton generally includes false segments that must be removed by a pruning step.

Applying a proper pruning method that would yield reliable centerlines is critical in all tree-skeletonization applications. An unwanted branch causes false generation numbering and consequently false measurements corresponding to the individual segments of the tree (including length, volume, surface area, etc.).

There are pruning approaches (e.g., morphological pruning⁸) capable of removing all side branches that are shorter than a predefined threshold. Those methods necessarily fail in structures consisting of tubular segments of varying thickness. Therefore, we have developed a method capable of removing “long” parasitic branches from “thick” parts and preserving “short” correctly determined branches in “thin” segments. Our pruning process consists of the following two phases:

- cutting holes that remain after the morphological closing, and
- deleting side branches using both the length and depth information.

At first, the centerlines are converted into a graph structure (each voxel corresponds to a graph node/vertex and there is an edge between two nodes if the corresponding voxels are 26-adjacent). Then, Dijkstra’s algorithm is applied to solve the single-source shortest-paths problem⁹ that maintains a rooted tree from a source node (i.e., the root detected in the first nonzero 2D slice in direction z). Since the result of Dijkstra’s algorithm is always a (cycle-free) tree, we can detect and cut holes in the centerlines easily: a skeletal point is to be deleted if it is not a line-end point and is not the parent of any other point in the Dijkstra’s tree. This heuristic hole-cutting approach works well, although counter-examples can be given in which the heuristic does not apply.

After the hole cutting, the parasitic side branches are removed. We have developed a centerline pruning that uses both the branch length and the distance-from-surface (depth) information for the identification of a pruning candidate. The distance-from-surface is calculated by linear time (3,4,5)-chamfer distance mapping¹. A branch is to be deleted if its length is shorter than a given threshold l and the value in the distance map corresponding to its branch-point (junction) is larger than a given threshold d . This pruning step can be repeated for different pairs of thresholds (l_i, d_i) . In our experience, 2 to 4 steps typically provide satisfactory results for in-vivo airway trees. The result of our pruning is demonstrated in the Fig. 3.

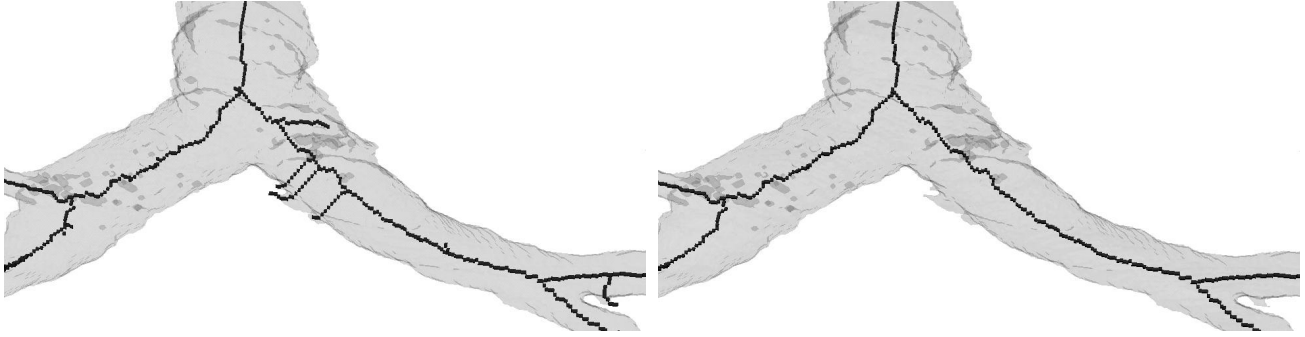


Figure 3. A part of a segmented tree and its centerline before pruning (left) and after pruning (right). The applied pruning technique can delete unwanted long branches from thick parts and unwanted shorter ones from thinner parts, while correct branches are typically preserved throughout the tree.

2.5. Branch-point Identification

In a skeleton, three types of points can be identified: end-points (which have only one 26-neighbor), line points (which have exactly two 26-neighbors), and branch-points (which have more than two 26-neighbors) that form junctions (bifurcations, trifurcations, etc.), see Fig. 4. Clearly, branch-identification from maximally thinned centerlines of an elongated tree is trivial. One problem is that more than one branch-points may form a junction. In that case, the branch-point closest to the root of the tree is assigned the junction label.

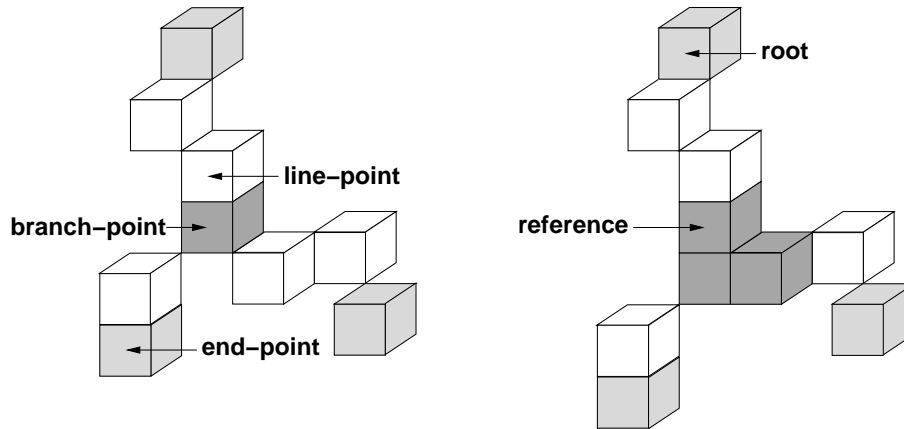


Figure 4. The three types of voxels in a maximally thinned structure (left). Sometimes, a junction is formed by more than one branch-points (right). In that case, the branch-point closest to the root is the reference point of the junction.

2.6. Generating a Formal Tree Structure

The formal tree structure assigned to the pruned centerlines is based on the updated Dijkstra's tree (after pruning). It is stored in an array of n elements for a centerline containing n voxels. Each element of that array stores the coordinates of a voxel, its depth in the elongated volume, and the index of the element that corresponds to the parent/predecessor voxel in the tree. This internal data structure is suitable for the forthcoming analysis and measurements, and provides an efficient coding of the resulted binary image. A similar structure is assigned to the branch-points. In the branch-tree, a path between two branch-points is replaced by a single edge.

2.7. Labeling

The aim of the labeling procedure is to partition all voxels of the binary tree into branches — each voxel is assigned a branch-specific label. There are two inputs into the process — the binary image after topological

corrections, and the formal tree structure corresponding to the centerlines. The output is a gray-level image, in which value “0” corresponds to the background and different non-zero values are assigned to the voxels belonging to different tree branches.

The automated labeling consists of two steps. First, only the voxels in the centerlines are labeled so that each branch-centerline has a unique label. Non-skeletal tree voxels are then labeled by label-propagation — each voxel in the tree gets the label of the closest skeletal point.

3. EXPERIMENTAL METHODS

3.1. Data

The reproducibility experiments were performed in 343 instances of a computer phantom and in a rigid plastic phantom CT-scanned under 9 orientations.

The computer phantom was designed by Kitaoka et al.¹⁰ as a 3-dimensional structural model of the human airway tree. The model consists of 125 elongated branches and its centerlines have 62 branch-points and 64 end-points (including the root of the tree). Note, that the true positions of the branch-points are known. The generated object is embedded in a $150 \times 150 \times 150$ binary array containing unit-cube voxels. Independently, the phantom was rotated in 5 degree steps between -15 and $+15$ degrees along all three axes (i.e., all the $7^3 = 343$ orientations (r_x, r_y, r_z) in the set $\{-15, -10, -5, 0, 5, 10, 15\}^3$ were investigated). The phantom in its neutral orientation can be seen in Fig. 5.

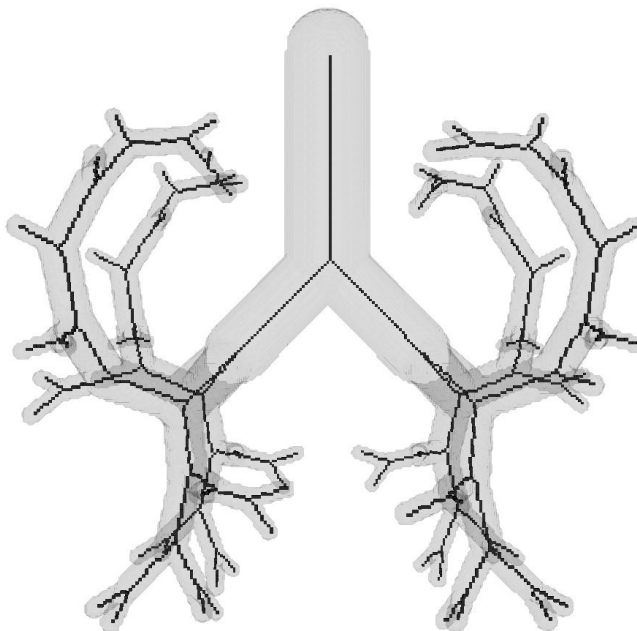


Figure 5. The $150 \times 150 \times 150$ computer phantom (in the neutral orientation) and its centerlines.

The second phantom is a hollow rigid plastic one, derived from an in-vivo scanned human bronchial tree, transformed in a computer graphics representation, and built by a rapid prototyping machine. It was imaged in 9 orientations using multi-row detector computed tomography with voxel size $0.439 \times 0.439 \times 0.5$ mm (4-slice spiral CT, Mx8000, Philips Medical Systems). The volume sizes were $512 \times 512 \times 300 - 400$ voxels. The rotation angles defined 9 phantom orientations in the scanner, the orientations were separated by 15° intervals in the $x - z$ and $y - z$ planes. From 3D CT phantom images, segmentation was performed to separate bronchial airways from the lung parenchyma. This phantom consists of more than 100 branches and 50 branch-points (see Fig. 6).

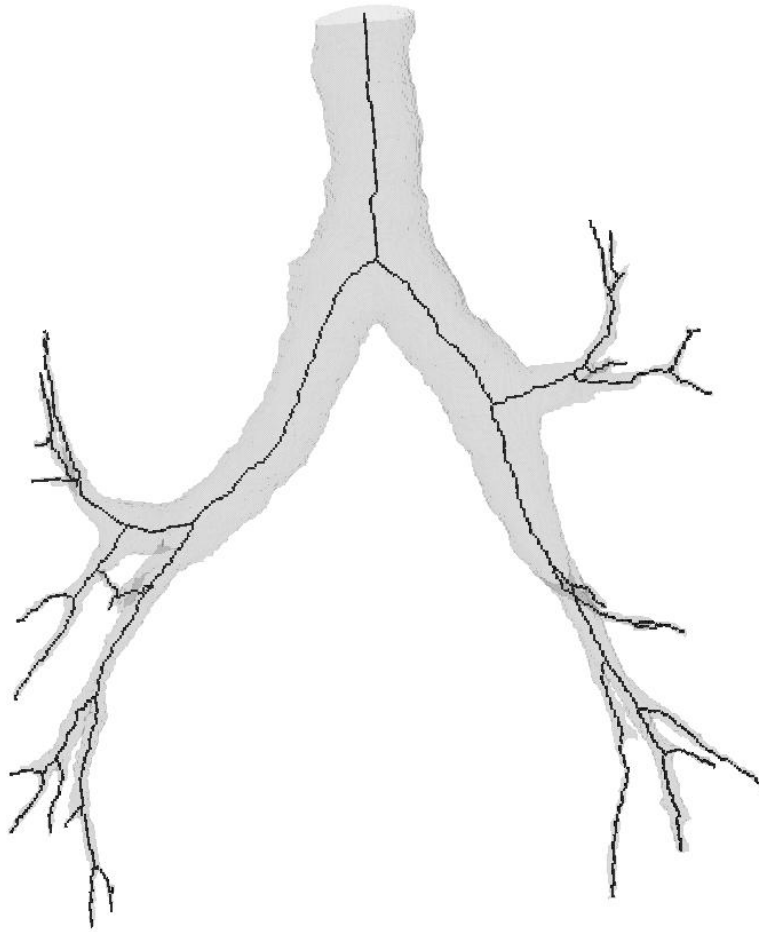


Figure 6. The rigid phantom (in the neutral orientation) and its centerlines.

3.2. Quantitative Indices

To evaluate the reproducibility of our airway tree skeletonization algorithm, the method described above was applied to the 3D binary images of airway trees. For each of the $343 + 9 = 352$ trees, skeletonization was performed fully automatically and the resulting skeletons were not interactively edited. For each instance of the computer phantoms, the branch-point position error was determined. It was defined as a Euclidean distance between the skeletonization-determined and true coordinates of the corresponding branch-points.

For a subset of 7 computer phantoms and the 9 rigid phantoms, the following quantitative indices were determined for the first 5 generations of the matched trees. Here, the reproducibilities were determined by assessing differences between the reference tree and the tree analyzed in different orientations, after registering the analyzed tree with the reference tree. The tree in neutral position (rotation $(0,0,0)$) was used as a reference tree:

- branch length – defined as a Euclidean distance between the parent and child branch-points,
- branch volume – defined as a volume of all voxels belonging to the branch,
- branch surface area – defined as a surface area of all boundary voxels belonging to the branch,
- average branch diameter – calculated from the distance map.

3.3. Statistical Validation

The reproducibility results are reported separately for each of the two phantom studies. The average branch-point positioning errors are only calculated for the computer phantom for which the branch-points extracted from the phantom in neutral position are considered as reference points. These errors are presented as mean \pm standard deviation and reported in voxels. All other reproducibility indices were compared using Bland-Altman statistic for which the average value of all corresponding measurements was used as an independent variable. The reproducibility showing 95% confidence intervals are presented in the form of Bland-Altman agreement plots¹¹.

4. RESULTS

The experiment was performed to assess the reproducibility of branch-point location using our approach. First, the branch-points were identified in a neutral phantom orientation by our method. Then, branch-points were identified in the new phantom positions. The branch-points identified in the neutral orientation were rotated in the same way. Consequently, for each phantom orientation, a set of independent-standard branch-points was available. The steps for a phantom orientation is sketched in Fig. 7. The Euclidean distance between corresponding branch-points showed sub-voxel accuracy of 0.788 ± 0.407 voxel size.

The reproducibility of the quantitative indices of the tree morphology are given in Figs. 8–9.

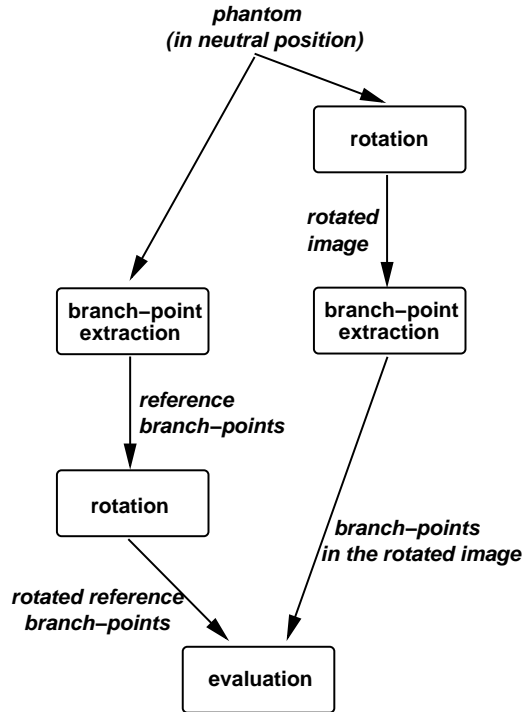


Figure 7. The steps for a phantom orientation. Note, that the same rotation angles were applied to the image data and the reference branch-points. The entire process was repeated 342 times.

5. DISCUSSION

Our algorithm for extracting centerlines from tree structures has several advantageous properties:

It produces geometrically correct centerlines due to the employed directional approach (i.e., the outmost layer of an object is peeled by 6 successive sub-iterations according to the 6 main directions). As a result, the centerline is in its correct position (i.e., in the middle of the object) and its location is fairly invariant under object orientation as discussed later.

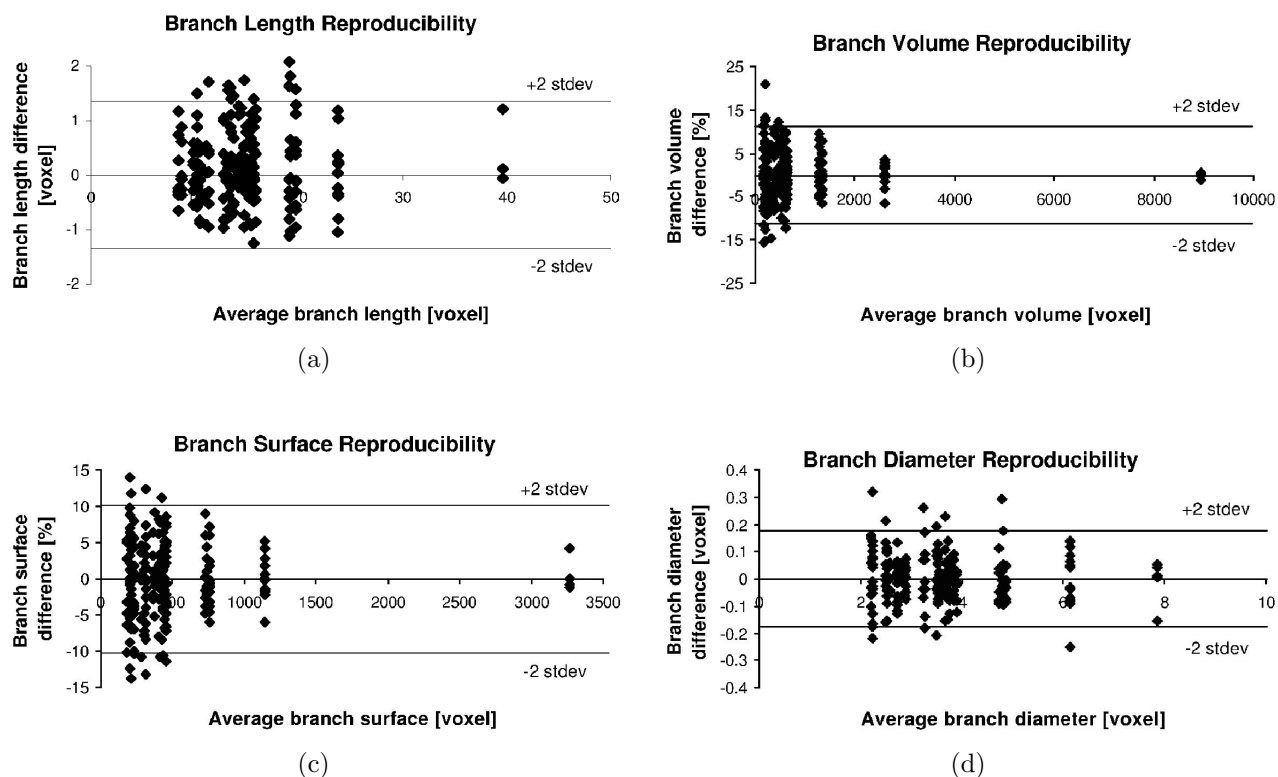


Figure 8. Reproducibility in a $150 \times 150 \times 150$ computer phantom. a) branch length, b) branch volume, c) branch surface area, d) average branch diameter.

The produced centerline is topologically equivalent to the original elongated object, since simple points are deleted sequentially. Our algorithm is topology-preserving by definition of simple points, therefore, the proof is self-evident.

The skeletonization algorithm can produce maximally thinned (i.e., 1-voxel wide) centerlines, since all simple points are deleted. Note, that some thinning algorithms may delete only a subset of simple points⁵. Therefore, the obtained structure is free from surface patches and any kinds of elongated parts. In comparison, the maximal thinness is not guaranteed by distance-based methods^{12, 13}.

Our skeletonization retains the shape of the original (elongated) object by preserving line-end points. The endpoint preserving thinning differs from shrinking (it is capable of extracting the topological kernel of an object, see Fig. 1). Our approach creates a substantially smaller number of unwanted centerline segments compared to competing methods⁷ due to a novel endpoint re-checking step.

Additionally, our method allows an easy and efficient implementation. Two linked lists are used; the first one stores all border points and the second one stores all border points of the actual type that are simple and not line-end points. The simplicity of a point is decided by determining a 26-bit integer code corresponding to their $3 \times 3 \times 3$ neighborhood and addressing a pre-calculated (unit time access) look-up-table containing the answers for all possible $3 \times 3 \times 3$ configurations. The codes of the marked points are stored in the second list. Therefore, the endpoint re-checking can be performed easily. Whenever a point is deleted, it is removed from the first list and its 6-neighbors (that are not in that list) are added. By using linked lists, the consumptive scanning of the volume is avoided. The proposed thinning algorithm is fairly fast; our implementation takes only 7 seconds for a $512 \times 512 \times 512$ image containing 250,000 object points voxels (running on a 1.7 MHz AMD Athlon 2000+ PC) including reading the input volume and the 8 MB look-up-table, and writing the output image.

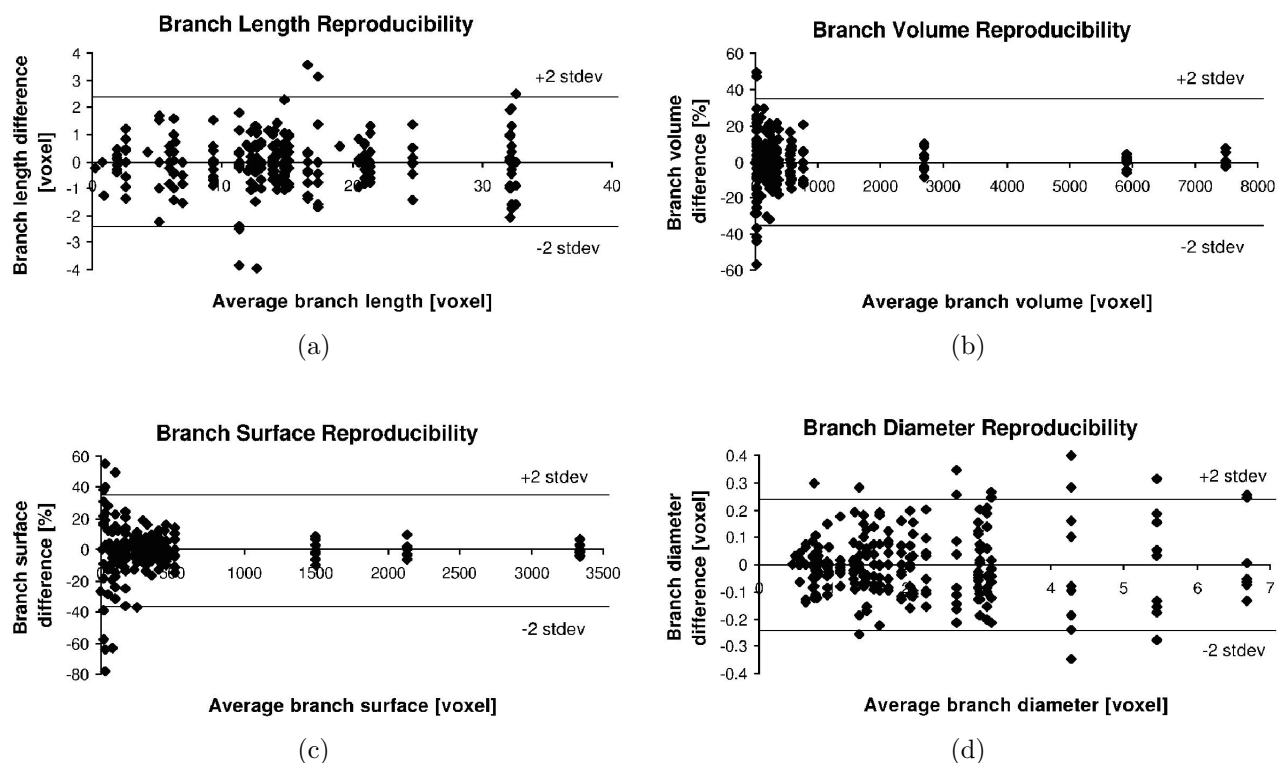


Figure 9. Reproducibility in a rigid phantom. a) branch length, b) branch volume, c) branch surface area, d) average branch diameter.

Our labeling method uses a generation-by-generation tree-walk strategy (similar to breadth-first-search⁹). Therefore, it can handle any kinds of tree including trees containing trifurcations, fourfurcations, etc. In comparison, the labeling method proposed by Mori et al. may fail if the trifurcation structure appears³. Note, that during the labeling process, an expanded branch-tree structure containing generation/layer numbers can be created. Additional quantitative indices including local diameter of a tree branch, bifurcation angle, etc. can be calculated.

6. CONCLUSION

An automated method for skeletonization, branch-point identification and quantitative analysis of tubular tree structures was presented. The method is robust, efficient, and highly reproducible.

ACKNOWLEDGMENTS

This work was supported by the NIH grant HL-064368.

REFERENCES

1. G. Borgefors, "Distance transformations in arbitrary dimensions," *Computer Vision, Graphics, and Image Processing* **27**, pp. 321–345, 1984.
2. M. Sonka, V. Hlavác, and R. Boyle, *Image Processing, Analysis, and Machine Vision — 2nd edition*, PWS, Boston, 1998.
3. K. Mori, J. Hasegawa, Y. Suenaga, and J. Toriwaki, "Automated anatomical labeling of the bronchial branch and its application to the virtual bronchoscopy system," *IEEE Trans. Medical Imaging* **19**, pp. 103–114, 2000.

4. T. Kong and A. Rosenfeld, "Digital topology: Introduction and survey," *Computer Vision, Graphics, and Image Processing* **48**, pp. 357–393, 1989.
5. K. Palágyi and A. Kuba, "A parallel 3d 12-subiteration thinning algorithm," *Graphical Models and Image Processing* **61**, pp. 199–221, 1999.
6. G. Malandain and G. Bertrand, "Fast characterization of 3d simple points," in *Proc. 11th IEEE International Conference on Pattern Recognition*, pp. 232–235, 1992.
7. T. Lee, R. Kashyap, and C. Chu, "Building skeleton models via 3-d medial surface/axis thinning algorithms," *CVGIP: Graphical Models and Image Processing* **56**, pp. 462–478, 1994.
8. R. Gonzales and R. Woods, *Digital image processing*, Addison-Wesley, Reading, Massachusetts, 1992.
9. T. Cormen, C. Leiserson, and R. Rivest, *Introduction to algorithms*, The MIT Press, 1990.
10. H. Kitaoka, R. Takaki, and B. Suki, "A three-dimensional model of the human airway tree," *Journal of Applied Physiology* **87**, pp. 2207–2217, 1999.
11. J. Bland and D. Altman, "Statistical methods for assessing agreement between two methods of clinical measurement," *Lancet* **1(8476)**, pp. 307–310, 1986.
12. C. Pudney, "Distance-ordered homotopic thinning, a skeletonization algorithm for 3d digital images," *Computer Vision and Image Understanding* **72**, pp. 404–413, 1998.
13. T. Saito and J. Toriwaki, "A sequential thinning algorithm for three dimensional digital pictures using the euclidean distance transformation," in *Proc. 9th Scandinavian Conf. Image Analysis, SCIA '95*, pp. 507–516, 1995.