Journal Code:				Article ID				Dispatch: 23.10.15	CE: Charlene A. Baylon
C	N	M		2	7	5	4	No. of Pages: 21	ME:

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN BIOMEDICAL ENGINEERING
Int. J. Numer. Meth. Biomed. Engng. (2015); e02754
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cnm.2754

Methods of graph network reconstruction in personalized medicine

A. Danilov[1,2], Yu. Ivanov[1,2], R. Pryamonosov[1,2,3] and Yu. Vassilevski[1,2,3,*,†]

[1]*Institute of Numerical Mathematics RAS, 119333, Moscow, 8 Gubkina St., Russia*
[2]*Moscow Institute of Physics and Technology, 141700, Moscow, 9 Institutskii Lane, Russia*
[3]*Moscow State University, 119991, Moscow, 1 Leninskie gory, Russia*

SUMMARY

The paper addresses methods for generation of individualized computational domains on the basis of medical imaging dataset. The computational domains will be used in one-dimensional (1D) and three-dimensional (3D)–1D coupled hemodynamic models. A 1D hemodynamic model employs a 1D network of a patient-specific vascular network with large number of vessels. The 1D network is the graph with nodes in the 3D space which bears additional geometric data such as length and radius of vessels. A 3D hemodynamic model requires a detailed 3D reconstruction of local parts of the vascular network. We propose algorithms which extend the automated segmentation of vascular and tubular structures, generation of centerlines, 1D network reconstruction, correction, and local adaptation. We consider two modes of centerline representation: (i) skeletal segments or sets of connected voxels and (ii) curved paths with corresponding radii. Individualized reconstruction of 1D networks depends on the mode of centerline representation. Efficiency of the proposed algorithms is demonstrated on several examples of 1D network reconstruction. The networks can be used in modeling of blood flows as well as other physiological processes in tubular structures. Copyright © 2015 John Wiley & Sons, Ltd.

Received 16 February 2015; Revised 6 October 2015; Accepted 9 October 2015

KEY WORDS: segmentation; skeleton; centerline; graph matching; hemodynamics; coronary arteries

1. INTRODUCTION

Personalized numerical modeling in biomedical applications has received a great deal of attention over many years, and a vast number of models have been described in the literature. Personalized techniques evaluate patient-specific model parameters and reconstruct patient-specific geometric model on the basis of medical imaging dataset. Given an imaging dataset, one performs image segmentation, volume reconstruction, and numerical discretization. In personalized hemodynamic modeling, two types of mathematical models are conventional: fluid dynamics models in three-dimensional (3D) domain [1–4] and fluid flows in one-dimensional (1D) network of collapsible tubes [5–10]. Both models have their pros and cons. Several 3D–1D coupling techniques are proposed to benefit from both models (rf. [11] and references therein). In this paper, we address personalized 1D network reconstruction methods which are needed in 1D hemodynamic simulations and 3D–1D coupled models. The proposed pipeline consists of vessel segmentation, centerline extraction, and 1D network reconstruction. Note that the segmented vessel image can be used for both centerline extraction and local 3D volume reconstruction required in 3D simulations.

In the present work, we assume that voxel-based medical image dataset is provided by protocols ensuring high contrast vessel images. The specific imaging procedures for contrast-enhanced magnetic resonance angiography and other medical imaging techniques lie outside the scope of this paper. Various medical image segmentation techniques have been developed [12–14].

*Correspondence to: Yu.Vassilevski, Institute of Numerical Mathematics RAS, 119333, Moscow, 8 Gubkina St., Russia.
†E-mail: yuri.vassilevski@gmail.com

Some approaches are computationally expensive and require parallel computing, in particular, the usage of graphic processing units [15]. In our paper, we consider low-cost segmentation algorithms, which do not require special hardware. The primary objective of our research is to develop fast, robust, and fully automatic segmentation of coronary arteries, critical importance of coronary vascular models being discussed in [16]. To this end, we use isoperimetric distance trees (IDT) algorithm, the fast variant of the isoperimetric algorithm [17], presented in [18] for aorta segmentation. Its robustness (more than 99% of segmentations are correct) [19] makes IDT-based algorithm more appealing than Hough circleness based approach [20]. Methods discussed in [19] extract centerlines and provide no segmentation, and we follow [20] and construct segmentation and skeletonization of coronary vessels by Frangi vesselness filter [21] and distance-ordered homotopic thinning [22]. Frangi filter is based on Hessian 3D analysis and performs segmentation of all tubular structures from the dataset. We should also note that the algorithms described in our paper may be applied to other tubular structures in human body such as airways and lymphatic system.

Centerlines and curve skeletons (or simply *skeletons*) are commonly used to reconstruct 1D networks. These structures are also used for accurate length measurements and local radius estimation. The concepts of a centerline or a skeleton are quite intuitive, although several mathematical definitions may be used. The first concept of a centerline was introduced by Blum [23]. Different methods were proposed for computing centerlines and skeletons, they use different definitions, and therefore, may produce different results [24–27]. Subject to application, a method can produce skeleton or centerline discontinuities that have to be fixed. An example of postprocessing algorithm eliminating network discontinuities can be found in [28], where skeletonization of microvasculature is provided by ImageJ and Amira software. Both centerlines and skeletons share the following common features: *connectivity*, *centredness* , and *1D dimensionality*. Connectivity requires the preservation of object compactness. If we use voxel representation for both the initial and the resulting objects, then centerline or skeleton of a compact shape is a set of connected voxels. Centredness requires that the resulting object should be locally centered within the input shape with respect to the shape boundary. 1D requires that the resulting object should be as thin as the representation permits, for example, for voxel data the resulting object should be one voxel thick. In our paper, we use two distinct definitions of a skeleton and a centerline, assuming that: a skeleton is a 1D topological structure with branches, while a centerline is a single path with no branches; a skeleton is uniquely determined by the input shape, whereas a centerline must be bounded by two given endpoints.

The problems of skeletonization and centerline extraction received a major attention over many years, and several algorithms have been proposed.

Voronoi diagram methods find centerlines as the paths in Voronoi diagrams that minimize the integral of the radii of the maximal inscribed spheres along the path. This method was implemented in the vascular modeling toolkit (VMTK) [29].

Distance mapping methods are generally used to construct the shortest path between two points. The first phase computes the distance from source point to each voxel inside the 3D object, called distance from source map. The shortest path from end point to the source point is found by descending through the gradient of the distance from source map. The centredness of the path may not be preserved using this technique. Similar approach may be used to compute skeletons using distance from boundary(DFB) fields [25]. The centredness problem may be solved by adding a penalty value to the distance cost at each node to keep shortest paths away from the boundary. This is effectively solved by employing the DFB field as node weights and extracting a minimum-cost spanning tree build from the DFB field [30].

Level set methods have been found to produce accurate centerlines [31]. Several voxel-based methods have been proposed to compute skeletons of 3D data [24, 25, 32, 33]. A representative comparison of skeleton techniques is given in [34]. Methods of centerline extraction without prior segmentation [35, 36] are outside the scope of the paper because 1D–3D coupled models require a local 3D vessel reconstruction.

Topological thinning methods remove voxels on the boundary of the shape while preserving connectivity and topology [22, 37, 38]. Voxel removal is performed in the order based on its distance to the boundary so that to enforce centredness feature.

DOI: 10.1002/cnm

In our work, we adopt the fully automatic topological thinning method [22]. We propose the essential postprocessing step in order to eliminate false twigs from the skeleton. The new algorithm automatically removes short twigs occurring commonly near bifurcations and flattened vessels. We discuss also the extraction of essential geometric information (radii, cross-section areas, and lengths of vessels) from thinning procedure data (distance map and skeleton). Alternative, more sophisticated, methods are considered in [20].

Moreover, we propose two algorithms for reconstruction of a 1D network either from skeleton or from a bunch of centerlines with a common endpoint. In the latter case, a centerline should be constructed for each vessel segment separately. We present several moderate size examples of 1D network reconstruction via centerline extraction using VMTK software [29] and manual endpoints placement. Along with 1D network graph structure, we also extract essential geometric information for each network segment. Minimal exported information consists of segment length and mean radius or cross-section area of the corresponding vessel. One can extract more detailed information: vessel radii at the ends of each segment, vessel radius, cross-section area, and curvature at each point of the 1D network.

Once we obtained the graph representation of the vascular domain, we can postprocess it. We discuss algorithms for graph matching introduced in [39] and address their applications to network reconstruction enhancement. In particular, we consider how a reference global graph can be adapted locally to obtain a patient-specific 1D network.

All methodologies that will be discussed later are summarized in Figure 1a. The algorithms can be organized in a processing chain in several ways subject to the type of input data. An example of fully automatic processing chain for coronary arteries network reconstruction is shown in Figure 1b. F1

The novelties of this paper are the following algorithms: fully automatic coronary artery segmentation, automatic reconstruction of 1D network using the skeleton extraction from a segmented image, automatic 1D network extraction from a set of centerlines with a common root, two 1D network graph postprocessing algorithms for graph correction, and local adaptation.

The outline of the paper is as follows. The segmentation algorithm for coronary arteries is presented in Section 1. The skeletonization algorithm is proposed in Section 2. In Section 3, we discuss techniques for graph construction. Section 4 introduces postprocessing techniques for graphs. Numerical results are presented in Section 5.

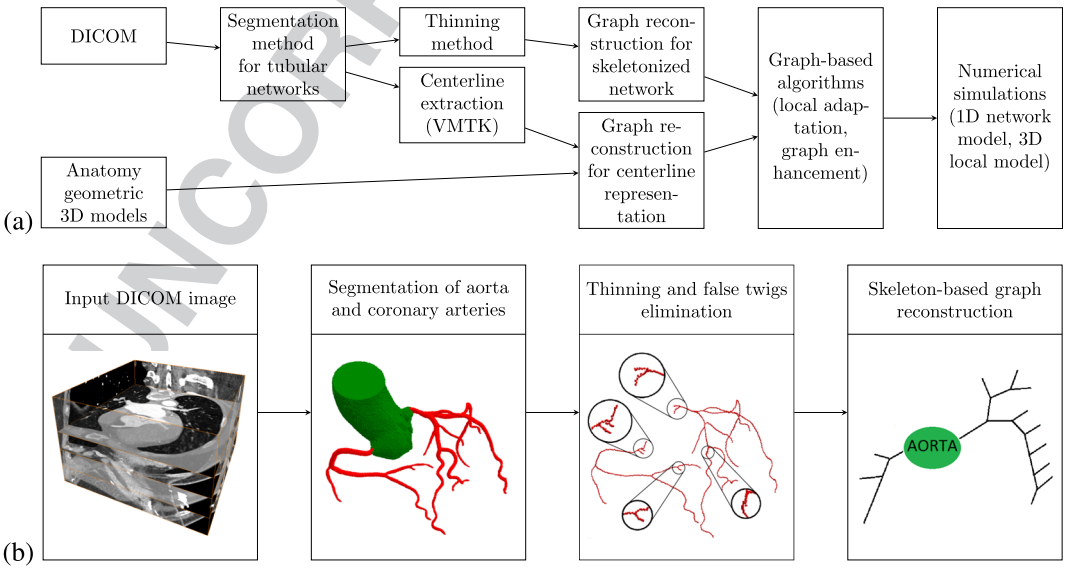


Figure 1. Admissible flowcharts of graph reconstruction algorithms (a) and flowchart for automatic reconstruction of coronary arteries 1D network (b).

DOI: 10.1002/cnm

2. SEGMENTATION METHODS FOR TUBULAR NETWORKS

The goal of segmentation is to assign material labels to voxels, the cells of a voxel grid. Hereafter, the voxel is assumed to be a cube or a box deviating from a cube by no more than 10 %. A voxel grid with more anisotropic cells must be resampled. The next definition is fundamental in this work and it is useful to give it in a very beginning. Given a voxel grid, we define binary mask (or mask) as assignment to each voxel 0 or 1. The 1-voxels compose foreground and form the object of interest, whereas 0-voxels compose background. A voxel is said to belong to a mask if it belongs to foreground.

One of the primary objectives of our research is the development of fast-fully automatic segmentation of coronary arteries. Our segmentation technology resembles algorithms [19, 20] and consists of three stages: aorta segmentation, ostia points detection, and segmentation of coronary arteries. The output data is a binary mask of vessels.

The first step in our automated segmentation algorithm for coronary arteries is obtaining the binary mask of aorta. Similarly to [19], our aorta segmentation algorithm is based on IDT method [18]. The IDT method is fast, robust, and provides anatomically correct results. It inputs an initial mask and a voxel from this mask, cuts a mask at bottlenecks, and outputs a submask of the initial mask containing the input voxel. The fully automated aorta segmentation is defined by the following algorithm:

Algorithm 1 Aorta segmentation.

1: Find an image with anatomically uppermost cut, compute the radius R_A , and the center $\mathbf{x}_{\text{center}}$ of the largest bright disk using the Hough circleness transform (Figure 2a).
2: Find the threshold T as the minimal intensity inside the disk.
3: Extract the connected mask M according to the threshold T (Figure 2b).
4: Apply the IDT to M and $\mathbf{x}_{\text{center}}$ and obtain the mask of aorta M_A (Figure 2c).
5: Eliminate from M_A voxels distanced from the boundary of M_A closer than a smoothing parameter R (Figure 2d).
6: Add to M_A voxels distanced from the boundary of M_A closer than the smoothing parameter R (Figure 2e).

F2 Algorithm 1 is illustrated in Figure 2. The largest bright disk at the first image correspond the uppermost aorta cut. Apart of the aorta, mask M contains contrast enhanced parts of the heart and other vessels, and therefore, the IDT algorithm is in demand.

For computation of distance to the boundary, we introduce the distance map: initiate all mask voxels $\mathbf{x}$ by

$$\text{dist}\,(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \notin \text{ mask}, \\ \inf & \text{if } \mathbf{x} \in \text{ mask}, \end{cases}$$

where inf is a large integer exceeding the total number of voxels; the distance map is defined recursively:

$$\text{dist}(\mathbf{x}) = \min_{\mathbf{x}_0 \in N_6(\mathbf{x})} (\text{dist}\,(\mathbf{x}_0) + 1),$$

where $N_6(\mathbf{x})$ denotes the set of voxels 6-adjacent to $\mathbf{x}$ (i.e. voxels sharing one common face with voxel $\mathbf{x}$). Note that at Step 6 of Algorithm 1, the distance map is computed for the background of M_A.

The last two steps of the aorta segmentation algorithm smooth the mask M_A and denoise its boundary. The noises may occur due to inappropriate choice of the threshold T and wrong assignment of coronary vessels parts to the mask of aorta at the IDT stage. Therefore, smoothing parameter R should not be less than the maximum radius $r_{\max}$ of coronary vessels and essentially less than R_A. In practice, instead of $r_{\max}$ computation, we set $R = 15 \geqslant r_{\max}$: for four CT and MRI datasets with different noisiness such choice provided successful aorta segmentations, Section 5.

DOI: 10.1002/cnm

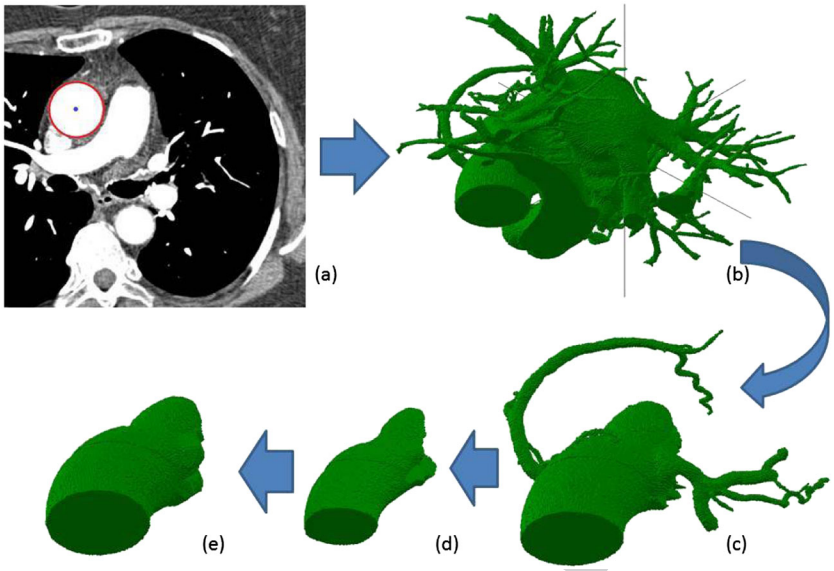


Figure 2. Aorta segmentation pipeline: (a) red circle with radius R_A and center $\mathbf{x}_{\text{center}}$ corresponds to aorta cut, (b) connected initial mask M containing $\mathbf{x}_{\text{center}}$, (c) output of IDT contains a noisy mask of aorta and parts of coronary vessels, (d) shrinking mask M_A at distance R (in voxel units), (e) expanding mask M_A at distance R (in voxel units) provides the aorta segmentation.

Q4

For vascular segmentation, we exploit Frangi vesselness filter. It is based on eigenvalues of the Hessian of an image function $L(\mathbf{x})$, where $\mathbf{x}$ is 3D coordinate vector. The filter measures similarity to a tubular structure and assigns vesselness value to each input voxel. Thresholding of the vesselness values provides a binary mask of vascular structures. The approach gives anatomically correct segmentation of all input voxels regardless of possible image discontinuities. The general idea of the local analysis of image $L(\mathbf{x})$ is based on its Taylor expansion. For computation of the Hessian $H_s(\mathbf{x})$ of L , one uses normalized derivatives in accordance with the linear scale space theory [40]:

$$\frac{\partial}{\partial x_i} L_s(\mathbf{x}) = s^\gamma L(\mathbf{x}) * \frac{\partial}{\partial x_i} G(\mathbf{x}, s),$$

where $*$ denotes the convolution, $G(\mathbf{x}, s) := \frac{1}{\sqrt{2\pi s^2}^3} \mathrm{e}^{\frac{-||\mathbf{x}||^2}{2s^2}}$ is the 3D Gauss kernel, $||\mathbf{x}||$ is the Euclidean norm of vector $\mathbf{x} \in \Re^3$. Hereinafter, the length is measured in voxels, that is, the voxel size is assumed to be equal to 1. Parameter s with length dimension defines the scale and has the following meaning: the second directional derivative of $G(\mathbf{x}, s)$ at scale s allows to gauge the contrast between the interior and exterior parts of the interval $(-s, s)$ along the direction. Parameter γ defines a family of normalized derivatives [41]. This normalization is important for a fair comparison of the response of differential operators at multiple scales. For each voxel vesselness $V(s)$ at scale s is computed via the eigenvalues $|\lambda_1| \ll |\lambda_2| \approx |\lambda_3|$ of the Hessian H_s:

$$V(s) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \text{ or } \lambda_3 > 0, \\ \left(1 - \exp\left(-\frac{\mathcal{R}_A^2}{2\alpha^2}\right)\right) \exp\left(-\frac{\mathcal{R}_B^2}{2\beta^2}\right) \left(1 - \exp\left(-\frac{\mathcal{S}^2}{2c^2}\right)\right), & \text{otherwise.} \end{cases}$$

Here, $\mathcal{R}_B = |\lambda_1|/\sqrt{|\lambda_2\lambda_3|}$ accounts the deviation from a blob-like structure, $\mathcal{R}_A = |\lambda_2|/|\lambda_3|$ is ratio distinguishing between plate-like and line-like structure, $\mathcal{S} = ||H||_F = \sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}$ is the Frobenius norm of H_s. Frangi parameters α, β, and c define thresholds that control the sensitivity of filter to measure $\mathcal{R}_A$, $\mathcal{R}_B$, and $\mathcal{S}$, respectively.

Segmentation of vessels with various radii is easily performed with the scale range $[s_{\min}, s_{\max}]$:

DOI: 10.1002/cnm

$$V = \max_{s_{\min} \leq s \leq s_{\max}} V(s).$$

In practice, vesselness filter detects both vessels and bones. Therefore, the Frangi filter segmentation of large vessels requires preprocessing of the input image, which obscures all bones. Heart imaging dataset have relatively low resolution, and the filter computation is not time consuming in this case.

Given the mask of aorta, we apply the Frangi filter with parameters $\alpha = 0.5$, $\beta = 0.5$, $c = 300$ for three scales $s = 1, 2, 3$. Ostia points are detected as two distinct local maxima of Frangi vesselness inside the mask of aorta. The coronary arteries are defined as components of vascular trees rooting at ostia points.

3. SKELETONIZATION OF SEGMENTED TUBULAR NETWORKS

Centerline is the convenient entity for description of tubular structures. There exist several methods of centerlines extraction without image segmentation. Because our objective is to generate 1D vascular networks and 3D reconstruction of local vascular regions, we shall not address these methods. Skeletonization is a fully automatic approach for centerline extraction. The thinning method generates vascular skeletonization. The input for the method is a binary mask of segmented object. The output is a skeleton (one-voxel thick set of connected voxels). The postprocessing step is used for automatic centerlines extraction and graph construction.

3.1. Thinning method

Among different implementations of the thinning idea, we adopt the distance-ordered homotopic thinning proposed by C. Pudney [22]. We use the definition of binary digital image (Z, F, m, n) from [42], where Z is a binary mask, F is a mask foreground, parameters m and n define the connectivity rules of m-adjacency for foreground voxels and n-adjacency for background voxels. The algorithm starts with a binary image (Z, F, m, n) and outputs the skeleton topologically equivalent to Z and centered with respect to the shape of Z. Set $N_{26}(p)$ denotes the set of voxels neighbouring to voxel p and sharing either a face, or an edge, or a vertex, this connectivity rule is called *26-adjacency*. Set $N_6(p)$ denotes the set of voxels neighbouring to voxel p and sharing a face, this connectivity rule is called *6-adjacency*. Voxel p is called *simple*, if it can be deleted from Z by setting $Z(p) = 0$ without topology modification of the shape of Z. The rule for determining whether voxel is simple is presented in [43]. The thickness of a skeleton is determined by m-adjacency and n-adjacency. In our work, we set $m = 26$ and $n = 6$.

The centredness of the skeleton is determined by the order of voxel elimination from Z. Following [22], we define a chamfer distance transform approximating Euclidean distance transform:

$$\text{dmap}\,(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \text{ is a background voxel,} \\ \min\limits_{\mathbf{x}_0 \in N_{26}(\mathbf{x})} (\text{dmap}\,(\mathbf{x}_0) + \text{dist}\,(\mathbf{x}, \mathbf{x}_0)) & \text{if } \mathbf{x} \text{ is a foreground voxel.} \end{cases}$$

The initial value for all foreground voxels $\mathbf{x}$ in this recursive definition is defined as $\text{dmap}(\mathbf{x}) = \text{inf}$, the value of inf is big enough (the threefold total number of foreground voxels) and $\text{dist}(\mathbf{x}, \mathbf{x}_0)$ is defined as

$$\text{dist}(\mathbf{x}, \mathbf{x}_0) = \begin{cases} n_1 & \text{if } ||\mathbf{x} - \mathbf{x}_0||^2 = 1, \\ n_2 & \text{if } ||\mathbf{x} - \mathbf{x}_0||^2 = 2, \\ n_3 & \text{if } ||\mathbf{x} - \mathbf{x}_0||^2 = 3. \end{cases}$$

Borgefors in his work [44] showed that setting $n_1 = 3$, $n_2 = 4$, and $n_3 = 5$ minimizes the upper bound on the difference between the chamfer and Euclidean distances. In the definition of $\text{dist}(\mathbf{x}, \mathbf{x}_0)$, we use Euclidean norm, assuming a voxel is a unit cube.

DOI: 10.1002/cnm

During the thinning process, the voxels are sorted by dmap values in ascending order and are deleted in groups with the equal values. In his work, Pudney defines a center maximal ball using the chamfer transform definition and introduces the two following *deletability* rules.

1. A voxel is deletable if it is simple and not the center of a maximal ball with distance value greater than a user-defined threshold.
2. A voxel is deletable if it is simple and not the end of a medial axis (i.e. it is 6-adjacent to one and only one foreground voxel).

In the original work, the first rule is used at the first stage of thinning process, and the second rule is used at the second stage. In practice, this approach leads to a skeleton with several false twigs, usually near bifurcations and flattened vessels, which do not correspond to any actual vessel. False twigs are attributed to superfluous topology features caused by initial image noises and irregularities of vessel shape.

We developed an alternative approach: we use Pudney's algorithm with the second deletability rule, and then we apply a postprocessing stage to eliminate false twigs. This technique allows us to identify vessel structure using a postprocessed skeleton.

3.2. False twigs elimination

As described previously, the skeletonization process preserves the network topology but may produce false twigs to be eliminated. For mathematical definition of false twigs, we will use the following notations. Two voxels are called *adjacent*, if one of them is 26-adjacent to another. Two voxels V and G belong to the same *connectivity component*, if a series of voxels $V_0, V_1, V_2, \ldots, V_n$ exists, such that $V_0 = V$, $V_n = G$, and $\forall i = 1, \ldots, n$ voxels V_i and V_{i-1} are adjacent. All voxels belonging to the same connectivity component defines the latter one. Skeleton voxel is called *inner*, if it has exactly two adjacent voxels. Skeleton voxel that is not inner is called *voxel-node*. Skeleton voxel with at most one adjacent voxel is called *end-node*. The connectivity component of inner voxels along with two bounding voxel-nodes defines the *skeletal segment*. The voxel-nodes of a skeletal segment are called *segment ends*. The *length* of a skeletal segment is defined as a number of voxels in the skeletal segment. Finally, a *false twig* is a skeletal segment, with one of the segment ends being an end-node, and the length of the skeletal segment is smaller than a user-defined threshold parameter.

We developed Algorithm 2 for false twigs elimination. It consists of the following steps. All skeletal segments with end-nodes as segment ends are identified. All segments with length smaller than the threshold are eliminated simultaneously. The skeleton thickness is enforced: voxel-nodes adjacent to eliminated skeletal segments are analyzed, and excessive voxel-nodes are eliminated. If at least one skeletal segment is eliminated, we start over. Importantly, all existing false twigs should be eliminated simultaneously, otherwise valid vessel branches may be accidentally eliminated (ref. Figure 3). **F3**

We describe the false twig elimination step in detail. First, we delete the skeletal segment keeping its end near the junction. This segment end is labeled as *red* voxel. If the red voxel is simple, then we remove it. Whenever the voxel-node is removed, we mark adjacent voxel-nodes as *blue* voxels. We use skeletonization rule of deletability for each blue vertex: it is deletable if it is simple and not end-node. This rule is not applicable to red voxels, because they are usually end-nodes.

Figure 4 shows three 2D examples of false twig elimination process. The first row represents the initial voxel configuration; the final state is presented in the second row. In all cases $MaxLenOfFalseTwig = 4$. Hatched voxels represent false twigs; voxel-nodes are marked by circles. In the first case (Figure 4a,d), the left segment is identified as false twig, and node 5 is marked red. Voxel-nodes 6, 7, and 8 will be marked blue after false twig elimination. Nodes 6 and 8 will be kept, because they are not simple. Node 7 will be deleted because it is simple and not endpoint. At the next stage, voxels 6 and 8 are inner voxels, and node 9 will be marked blue (Figure 4d). Node 9 is kept, because it is not simple. In the second case (Figure 4b,e), left, right, and upper segments are false twigs and nodes 5, 6, and 7 are marked red. These nodes will be deleted, because they are simple. Voxel-nodes 8 and 9 will be marked blue. Both will be kept, because node 9 is not simple, **F4**

DOI: 10.1002/cnm

Color Online, B&W in Print

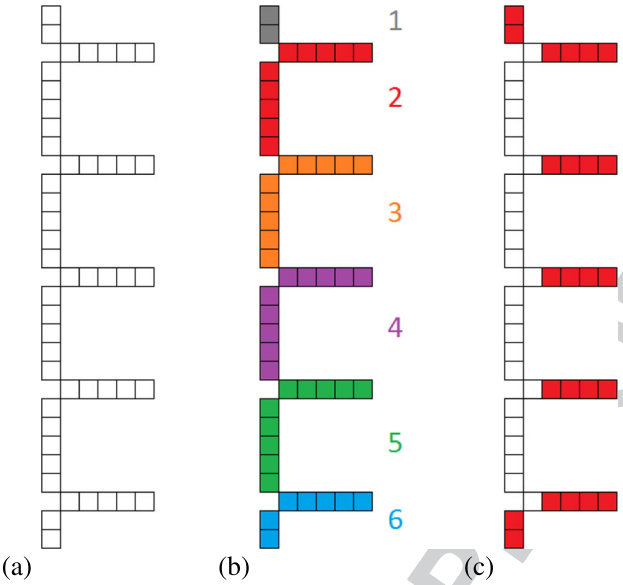


Figure 3. 2D example of false twigs elimination with threshold parameter set to 11 voxels: (a) initial skeleton, (b) consecutive order of false twigs elimination leads to elimination of all voxels, and (c) simultaneous false twigs elimination leads to accurate skeleton, which is colored white.

Color Online, B&W in Print

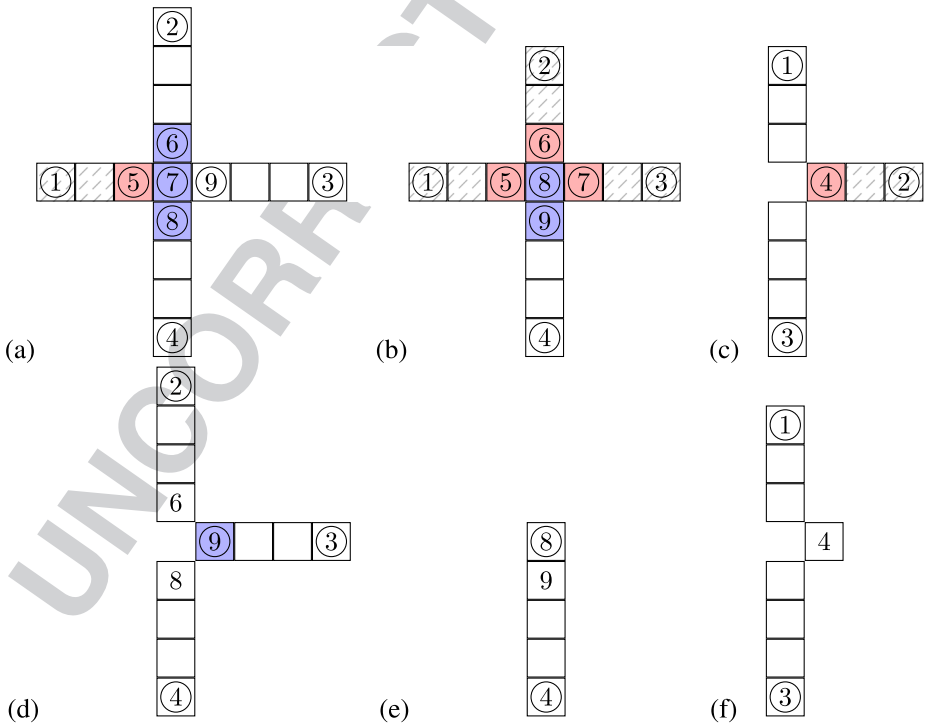


Figure 4. Three cases of false twigs elimination with threshold parameter set to four voxels (2D examples). The first row represents the initial voxel configuration; the final state is presented in the second row. Hatched voxels represent false twigs, voxel-nodes are marked by circles.

DOI: 10.1002/cnm

and node 8 is an end-node. At the final stage, voxel 9 is an inner voxel (Figure 4e). In the third case (Figure 4c,f), right segment is a false twig, and node 4 is marked red. This node will be kept, because it is not simple. At the final stage, voxel 4 is an inner voxel (Figure 4f).

Algorithm 2 False twigs elimination.

```
1: Set Nodes = ∅, FalseTwigs = ∅, RedNodes = ∅, BlueNodes = ∅
2: Find all voxel-nodes and push them into Nodes
3: for all end-nodes V ∈ Nodes do
4:    identify skeletal segment E with segment end V
5:    if length of E is smaller than MaxLenOfFalseTwig then
6:       push E into FalseTwigs
7:    end if
8: end for
9: if FalseTwigs is empty then
10:   exit the algorithm
11: end if
12: for all E ∈ FalseTwigs do
13:   eliminate all inner voxels of segment E from skeleton
14:   for all segment ends V of E do
15:      if V is end-node then
16:         eliminate V from skeleton
17:      else
18:         push V into RedNodes
19:      end if
20:   end for
21: end for
22: while RedNodes ≠ ∅ do
23:   pop voxel V from RedNodes
24:   if V is simple voxel then
25:      eliminate V from skeleton
26:      push into BlueNodes all adjacent to V voxels from Nodes
27:   end if
28: end while
29: while BlueNodes ≠ ∅ do
30:   pop voxel V from BlueNodes
31:   if V is simple voxel then
32:      eliminate V from skeleton
33:      push into BlueNodes all adjacent to V voxels from Nodes
34:   end if
35: end while
36: restart from line 1
```

The iterative Algorithm 2 eliminates all false twigs keeping intact voxels corresponding to vessel center lines. New false twigs may appear only in places where previous false twigs were deleted. Because no long skeletal segment is split during the process, no new twigs may appear. The described algorithm may be modified to skip previously examined valid skeletal segments. In this case, the algorithm will visit each skeletal segment only once. In practice, only few iterations of the algorithm are needed. The run time for this algorithm is presented in Table II, in row false twigs elimination.

In our algorithm, the threshold parameter MaxLenOfFalseTwig is fixed. Some tiny vessels with small radius may be shorter then MaxLenOfFalseTwig. We propose to use heuristic rule of local threshold adaptation: for each skeletal segment E, we set $\mathrm{MaxLenOfFalseTwig}(E) = 2r(V)$,

DOI: 10.1002/cnm

where $r(V)$ is the local radius in the corresponding red voxel V. The algorithm for $r(V)$ evaluation is presented in Section 3.1.

4. GRAPH STRUCTURE OF PREPROCESSED TUBULAR NETWORKS

Graph structures are commonly used to represent 1D tubular networks in different applications. In this section, we discuss two techniques for graph extraction from image skeleton and a set of centerlines with common root. Additional geometric information associated with graph is recovered as well: coordinates of graph nodes and mean radius of corresponding tubular segment for each graph edge.

4.1. Graph reconstruction for skeletonized networks

Once the skeleton is cleared from false twigs, the 1D graph reconstruction is straightforward. Each skeletal segment corresponds to a centerline of a vessel segment. The centers of voxels along the skeletal segment define the centerline of the segment. We need to compute the length of the line, as well as the mean vessel radius.

The following heuristic estimation is used for line length:

$$\mathrm{len}(V_0, V_1, \ldots, V_{N-1}) = ||V_{N-1} - V_{[(N-1)/S]\cdot S}|| + \sum_{i=1}^{[(N-1)/S]} ||V_{i\cdot S} - V_{(i-1)\cdot S}||,$$

where N is the length of the skeletal segment, V_i are the coordinates of $(i+1)$-th voxel along the segment. If a skeletal segment zigzags, the computed length across all voxels exceeds the actual length. For more accurate estimation, we count every S-th voxel in line length estimation. The bigger parameter S is used, the less sensitive to zigzags is the segment length. Basing on the comparisons with the straightforward computation of line lengths by VMTK tools, we set $S = 4$ to minimize the error of the estimation. Note that one can also use a robust approach based on multi-planar reformatted (MPR) images [20].

For vessels with circular cross-sections local vessel radius $r(V)$ in voxel V can be estimated from the chamfer distance transform via Algorithm 3.

Algorithm 3 Computation of vessel radius $r(V)$

Require: dmap
1: $U := V$
2: **while** dmap$(U) > 0$ **do**
3: $\quad U := \arg \min_{\mathbf{x} \in N_{26}(U)} \mathrm{dmap}(\mathbf{x})$
4: **end while**
5: **return** $||U - V||$

The quickest descent through chamfer transform map is used to find the nearest background voxel. The returned value (computed in voxel units) in fact is slightly bigger than the radius of the maximal inscribed ball with the center V. This value is used as estimation of the local vessel radius in voxel V. The mean vessel radius is computed as the mean value of local radii across the vessel.

For vessels with oblate elliptic cross-sections, we propose an MPR-based method for computation of mean cross-section area (Algorithm 4). Details on MPR image construction and detection of lumen contours can be found in [20]. Detection of lumen contours in step 3 is essential, because it reduces overestimation of the cross-section area near bifurcations. Longitudinal cuts can be constructed from spanning planes for $(\hat{e}_1, \hat{e}_2)$ and $(\hat{e}_1, \hat{e}_3)$ for every voxel in centerline, where $\hat{e}_i$ is the eigenvector corresponding to eigenvalue λ_i, $i = 1, 2, 3$, rf. Section 1. The formula in step 7 provides reasonable results for elliptic-shaped vessel cross-sections, because $\hat{e}_2, \hat{e}_3$ are oriented along semi-axes of an ellipse approximating the cross-section.

DOI: 10.1002/cnm

Algorithm 4 Computation of mean vessel cross-section areas along skeletal segment

Require: Skeletal segment, original DICOM dataset

1: Compute MPR image for skeletal segment
2: Find 2 longitudinal cuts L_1 and L_2 of MPR image
3: For each cut detect lumen contours
4: **for all** voxels V in skeletal segment **do**
5: Find 2D slice Π_V of MPR image containing voxel V
6: Compute from lumen contours the lumen diameters d_i in $\Pi_V \cap L_i, i = 1, 2$
7: Compute mean area for voxel V: $A_V = \frac{\pi}{4} d_1 d_2$
8: **end for**

The connectivity component of voxel-nodes is used to define the graph node coordinates. The latter are coordinates of the barycenter of the connectivity component of the voxel-nodes. Each skeletal segment has two associated segment ends that belong to some voxel-nodes components, which in turn correspond to graph nodes. This information is used to specify the connectivity of the graph.

4.2. Graph reconstruction for centerlines

In this section, we present technique for merging several centerlines in a 1D graph representing a tubular network. We assume that the centerlines were extracted either from a segmented image, or from a realistic 3D anatomy model, for example, from anatomy models from Plasticboy Pictures CC [45]. The minimum cost path construction implemented in VMTK software[29] generates for each vessel segment a centerline connecting two corresponding extremal points. We present in Section 5 several moderate size examples of VMTK 1D network reconstruction with manual endpoints placement.

We can represent each centerline C as an ordered set of pairs $C = \{(\mathbf{a}_i, r_i)\}_{i=1}^{n}$ and associate this centerline with graph edge $e = (\mathbf{a}_1, \mathbf{a}_n)$. The set of input centerlines is treated as a set of standalone graphs consisting of one edge. In order to merge those graphs in one network, we need to construct all edge intersections in each graph. Spatial position of tubular structures defined by centerlines is used for intersection evaluation.

We define *branching point* $\mathbf{a}_{ij}$ for two edges of centerlines $C' = \{(\mathbf{a}'_i, r'_i)\}_{i=1}^{n}$ and $C'' = \{(\mathbf{a}''_j, r''_j)\}_{j=1}^{m}$ based on the following condition (branching point is marked red in Figure 5b): **F5**

$$\exists \mathbf{a}'_i \in C', \mathbf{a}''_j \in C'' : \|\mathbf{a}'_i - \mathbf{a}''_j\| < r'_i + r''_j \Rightarrow \mathbf{a}_{ij} = \left(\mathbf{a}'_i + \mathbf{a}''_j\right)/2, r_{ij} = \left(r'_i + r''_j\right)/2, \quad (1)$$

where $\|\mathbf{x} - \mathbf{y}\|$ is an Euclidean distance between points $\mathbf{x}$ and $\mathbf{y}$.

The definition (1) is based on the assumption that the distance between centerline points is small enough: $\forall \mathbf{a}_i, \mathbf{a}_{i+1} \in C : \|\mathbf{a}_{i+1} - \mathbf{a}_i\| < \max(r_{i+1}, r_i)$. We can always enforce this condition by inserting additional points in the centerline.

Once a branching point is detected, we add new graph edges $(\mathbf{a}'_1, \mathbf{a}_{ij})$, $(\mathbf{a}_{ij}, \mathbf{a}'_n)$, $(\mathbf{a}'_1, \mathbf{a}_{ij})$, $(\mathbf{a}_{ij}, \mathbf{a}''_m)$ and remove previous edges $(\mathbf{a}'_1, \mathbf{a}'_n)$, $(\mathbf{a}''_1, \mathbf{a}''_m)$ merging two centerlines (Figure 5). The new edges store the original centerline information.

In general case, a pair of edges may have multiple branching points and additional considerations should be taken. In Figures 6a,b, different layouts of centerline intersections are presented. In all cases, merging of several adjacent branching points in one cluster is preferred. We define an auxiliary graph $G_c = (V_c, E_c)$, where branching points define the set of nodes V_c , and edges satisfy the following condition: **F6**

$$(\mathbf{a}_{ij}, \mathbf{a}_{kl}) \in E_c \Leftrightarrow \|\mathbf{a}_{ij} - \mathbf{a}_{kl}\| < r_{ij} + r_{kl}.$$

The set of branching points A may be split into subsets (clusters) $A_1, A_2, .., A_k$ using one of the following two rules:

DOI: 10.1002/cnm

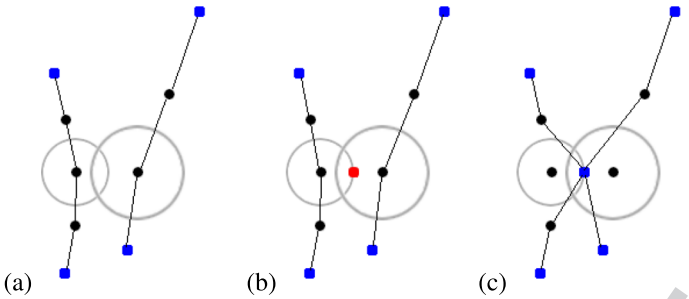


Figure 5. Branching points creation: (a) pair of edges with $\mathbf{a}'_i$ and $\mathbf{a}''_j$ complying with condition (1), (b) branching point $\mathbf{a}_{ij}$ marked red, and (c) modified graph after merging.

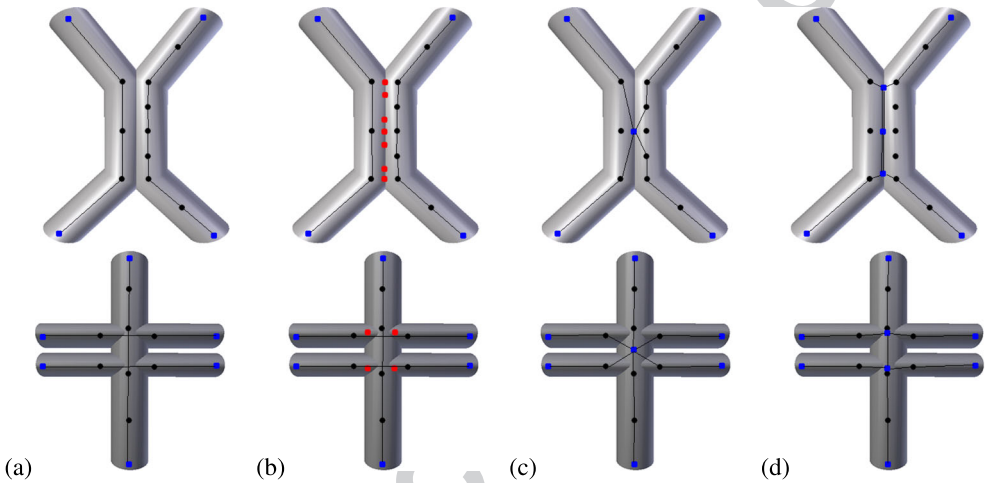


Figure 6. Different techniques for merging centerlines: (a) centerlines corresponding to several intersecting branches, (b) branching points (red dots) defined by condition (1), (c) and (d) nodes union into the branching points (blue dots) under Rules 1 and 2, respectively.

Rule 1: $\forall \mathbf{a}_i \in A_m \quad \exists \mathbf{a}_j \in A_m : \ (\mathbf{a}_i, \mathbf{a}_j) \in E_c$;
Rule 2: $\forall \mathbf{a}_i, \mathbf{a}_j \in A_m \quad (\mathbf{a}_i, \mathbf{a}_j) \in E_c$ and $\nexists \mathbf{a}_k \in A \setminus A_m : \ \forall \mathbf{a}_i \in A_m : \ (\mathbf{a}_k, \mathbf{a}_i) \in E_c$.

According to Rule 1, the set A_i corresponds to nodes from the i-th connected component of the graph G_c. According to Rule 2, each set corresponds to a maximum graph clique. A clique in graph G is a subgraph $C = (V_C, E_C) \subseteq G$ such that for every two nodes in V_C there exists an edge in E_C connecting them. A maximum clique is a clique with the largest number of nodes in the given graph. An algorithm implementing Rule 2 is presented in Algorithm 5. The algorithm for maximum clique search is based on [46] and in our work was implemented with NetworkX library.

Algorithm 5 Branching points clustering using Rule 2.

```
Set i = 1
Set G = G_c
while G ≠ ∅ do
    Find maximal clique K_max of graph G
    Set A_i as nodes from K_max
    G := G \ K_max
    i := i + 1
end while
```


DOI: 10.1002/cnm

Once the clusters are built, their centers are used as branching points, and the merging technique proposed previously is used. The result of clustering using both proposed rules is presented in Figure 6. We prefer to use Rule 2 because it produces more accurate results. This rule is used in numerical examples presented in Section 5.

We defined the algorithm for merging two single edge graphs in a graph network. In general case, two graph networks can be merged by applying this algorithm pairwise when the first edge of a pair belongs to the first graph, and the second edge belongs to the second graph. This technique is also applied when network graphs from different segmented images should be stitched together. The proposed technique is applicable to graphs with loops and multiple edges. If the tubular centerline is discontinued and the gap is less than the tubular radius, the proposed technique connects both centerlines automatically.

5. GRAPH-BASED APPLICATIONS

In Section 3, we considered graph reconstruction for a tubular network. Every graph edge corresponds to a tubular segment described by a centerline. Such graphs provide the base for the solution of practical problems in personalized 1D hemodynamic modeling.

First, the network reconstruction according to segmented *dynamic* datasets (e.g. coronary computed tomography) needs matching and coupling of structures extracted at different time moments. Vessel displacements and uneven distribution of contrast matter in blood make this task complicated: graph structures at different moments can be non-isomorphic and have some geometric differences.

Second, personalized 1D-hemodynamic modeling often requires global network reconstruction given regional data. Conventionally, there is no possibility to acquire an image dataset for the full vasculature of the patient. Moreover, only regions with pathologies can have computational significance in simulation. One can produce a generic network locally adapted to patient diagnostic data.

Additionally, having a reference graph, we can achieve significant enhancement of the automatic personalized reconstruction. The reference graph may be obtained from a thoroughly segmented volume of interest or be a manually created structure (e.g. with anatomic atlases). Association between the reference graph and a personalized graph can correct ruptures and other faults emerged from low-quality segmentation of patient vasculature.

These problems require association technique between elements of the graph. Standard algorithms dealing with isomorphic graphs are not applicable in these cases. We propose to use the alternative approach introduced in [39, 47]. It considers inexact graph matching on the basis of similarity between geometric and topological features of tubular structures.

5.1. Automatic graph matching method

Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs with sets of nodes V_1, V_2 and sets of edges E_1, E_2. The association graph $G_A = (V_A, E_A)$ of G_1 and G_2 is defined in two steps in accordance with the sets of functions S_V and S_E specifying similarity between nodes and edges, respectively. Both functions $f_i \in S_V$ and $g_i \in S_E$ vary in range [0, 1]. The closer function value is to 1 , the stronger similarity between elements is:

$$V_A = \left\{ v_a \in V_1 \times V_2 \mid \sum_{f_i \in S_V} \omega_i f_i(v_a) \geqslant \Delta_V \right\}, \quad \omega_i \in [0,1], \ \sum_i \omega_i = 1, \tag{2}$$

$$E_A = \left\{ (v_a, v_b) \in V_A \times V_A \mid \sum_{g_i \in S_E} \nu_i g_i(v_a, v_b) \geqslant \Delta_E \right\}, \quad \nu_i \in [0,1], \ \sum_i \nu_i = 1, \tag{3}$$

where Δ_V and Δ_E are threshold values in range [0, 1]. At the first step, we add to V_A nodes according to (2). This means that V_A corresponds to the set of all potential assignments of nodes in the

DOI: 10.1002/cnm

set V_1 to nodes in the set V_2. In order to determine whether a node $v_a = (v_{a_1}, v_{a_2}) \in V_A$ represents a promising assignment, the set of node similarity functions S_V must be computed. At the second step, we add to E_A edges according to condition (3). Two nodes $v_a = (v_{a_1}, v_{a_2}) \in V_A$ and $v_b = (v_{b_1}, v_{b_2}) \in V_A$ are connected by an edge if both two potential assignments are mutually consistent.

The choice of functions $f_i \in S_V$ and $g_i \in S_E$ depends on application. Functions $f_i \in S_V$ can be based on local properties of the network nodes:

1. Euclidean distance between nodes coordinates;
2. nodes degree or level (in the sense of the graph theory);
3. similarity between local radii of input/output branches;
4. similarity between shortest distances to root nodes.

Functions $g_i \in S_E$ can be constructed to catch the following features:

1. Euclidean distance between pair of nodes;
2. similarity of lengths along the centerlines $p(v_{a_1}, v_{b_1})$ and $p(v_{a_2}, v_{b_2})$;
3. similarity between curved centerlines $p(v_{a_1}, v_{b_1})$ and $p(v_{a_2}, v_{b_2})$;
4. similarity between local radii of corresponding centerlines.

Given an association graph G_A for G_1 and G_2, we can modify G_1 and G_2 by replacing and merging their elements. For particular applications we refer to Section 5.

5.2. Local adaptation of reference graph to diagnostic data

Let graph $B = (V_B, E_B)$ correspond to a reference global network structure, graph $C = (V_C, E_C)$ be reconstructed from patient pre-segmented CT/MRI data in a region of interest. In order to simulate the global blood flow in a global individualized 1D network, we need to acquire graph D by local adaptation of B to C.

At first, we find the association graph $G_A = (V_A, E_A)$. Let E_v denote the set of edges adjacent to v , and $r_v(e)$ denote the local radius of tube corresponding to e adjacent to v. We define functions $R_1(v)$, $R_2(v)$, $R_3(v)$:

$$R_1(v) = \max_{e \in E_v}(r_v(e)), \quad e_1 = \arg\max_{e \in E_v}(r_v(e)),$$

$$R_2(v) = \begin{cases} \max\limits_{e \in E_v \setminus e_1}(r_v(e)) & \text{if } E_v \setminus e_1 \neq \varnothing, \\ 0 & \text{if } E_v \setminus e_1 = \varnothing, \end{cases} \qquad e_2 = \arg\max_{e \in E_v \setminus e_1}(r_v(e)),$$

$$R_3(v) = \begin{cases} \max\limits_{e \in E_v \setminus (e_1 \cup e_2)}(r_v(e)) & \text{if } E_v \setminus (e_1 \cup e_2) \neq \varnothing, \\ 0 & \text{if } E_v \setminus (e_1 \cup e_2) = \varnothing, \end{cases} \qquad e_3 = \arg\max_{e \in E_v \setminus (e_1 \cup e_2)}(r_v(e)).$$

Function f_b determines similarity between branching nodes considering three adjacent edges of the maximum radius:

$$f_b(v_a) = f_b(v_1, v_2) = \prod_{i=1}^{3}\left(1 - \frac{|R_i(v_1) - R_i(v_2)|}{\max(R_1(v_1), R_1(v_2))}\right). \tag{4}$$

We use this function in (2) with weight $\omega_1 = 1.0$ and $\Delta_v = 0.65$ to find V_A. Then we determine set of edges E_A as edges of all possible shortest paths in graph B between $v_1, v_2 \in V_A$. To define the locally adapted graph D, we remove the set of edges E_A from E_B and replace it by E_C just merging it via association for V_A.

Importantly, functions f_b do not take into account spatial layout or orientation of graphs B and C, and preliminary alignment of graphs is not required. This may be useful for defining association between networks constructed from different datasets (e.g. for different patients) corresponding to the same anatomical regions with anthropometric similarity.

DOI: 10.1002/cnm

5.3. Graph matching of tubular network dynamic sequence

Let G_1 and G_2 be graphs reconstructed from the segmentations of the same region and distinguished by small spatial displacements and topological differences. This is typical for the coronary CT datasets or the usage of different tomography modes for the same patient diagnosis. In order to acquire more comprehensive reconstruction of the examined region, one can generate an association graph G_A that involves nodes and edges from G_1 and G_2. To acquire G_A, it suffices to define functions $f(v_a)$ and $g(v_a, v_b)$ from (2), (3). Parameters Δ_V, Δ_E should be close to 1 and subject to application.

Because of small spatial displacements, we can use Euclidean distance to determine similarity between nodes: $f(v_a) = f(v_1, v_2) = (1 - \|\mathbf{v}_1 - \mathbf{v}_2\|)$, where $\|\mathbf{v}_1 - \mathbf{v}_2\|$ is normalized in the range [0,1]. We use the dynamic time warping algorithm to determine similarity $g(v_a, v_b) = g((v_{a_1}, v_{b_1}), (v_{a_2}, v_{b_2}))$ between two centerlines $p(v_{a_1}, v_{b_1})$, $p(v_{a_2}, v_{b_2})$. We define recursively a distance function $D[i_1, i_2]$ between i_1-th point in centerline $p(v_{a_1}, v_{b_1})$ and i_2-th point in centerline $p(v_{a_2}, v_{b_2})$ according to (5):

$$D[i_1, i_2] = \|\mathbf{v}_{i_1} - \mathbf{v}_{i_2}\| + \min(D[i_1 - 1, i_2], D[i_1, i_2 - 1], D[i_1 - 1, i_2 - 1]). \tag{5}$$

We set $g((v_{a_1}, v_{b_1}), (v_{a_2}, v_{b_2})) = (1 - D'[n_1, n_2])$, where n_1 and n_2 are numbers of points in $p(v_{a_1}, v_{b_1})$ and $p(v_{a_2}, v_{b_2})$, respectively, and D' is normalized function D in the range [0,1]. For the detailed description of the dynamic time warping method, we refer to [48].

6. NUMERICAL RESULTS

The proposed segmentation, skeletonization, and graph construction algorithms were evaluated on two cardiac contrast-enhanced CT DICOM images (Figure 7) and micro-CT images of vascular corrosion cast of rabbit kidney (Figure 8). The resolution of cardiac images as well as run times for different segmentation steps are presented in Table I. The time for ostia points detection is negligibly small, and therefore is not included in the table. The most CPU time is attributed to Frangi filtering. The quality of provided CT images is quite high, thus no ruptures appear in the segmented images. A small number of false twigs appear mainly near the ostia points. F7 F8 T1

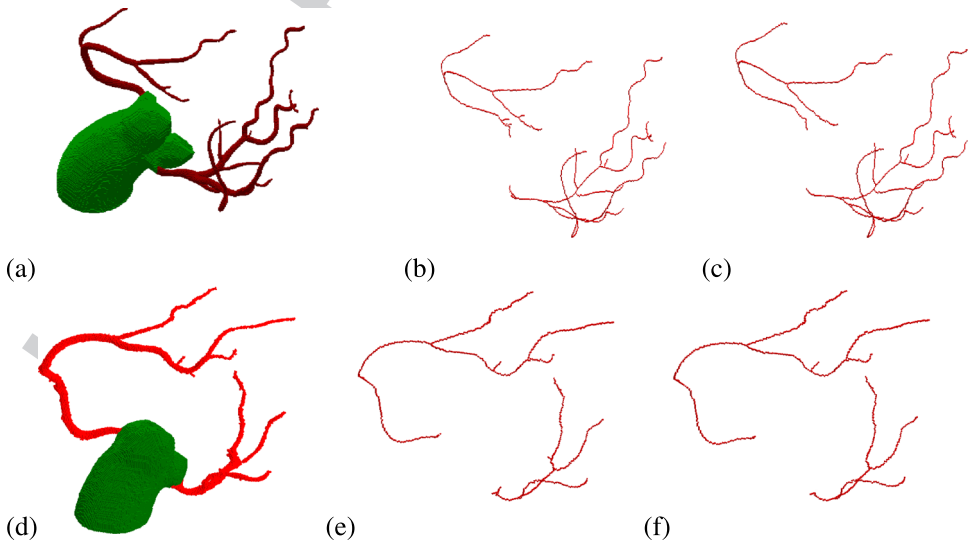


Figure 7. Segmentation and skeletonization of two cardiac CT images: (a) and (d) segmented aorta (green) and coronary arteries (red) for Case 1 and Case 2, respectively, (b) and (e) skeletons before false twigs elimination, (c) and (f) skeletons after false twigs elimination.

DOI: 10.1002/cnm

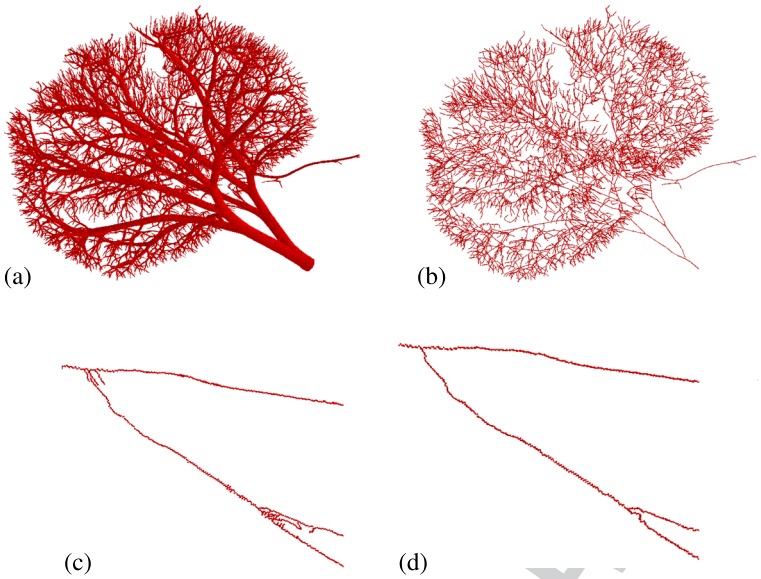


Figure 8. Segmentation and skeletonization of rabbit kidney: (a) segmented rabbit kidney, (b) skeleton after false twigs elimination, (c) bifurcations of skeletonized network before false twigs elimination, and (d) after false twigs elimination. μCT DICOM data was provided by J. Alastruey [49, 50].

Table I. Data set resolution and CPU time of coronary segmentation stages.

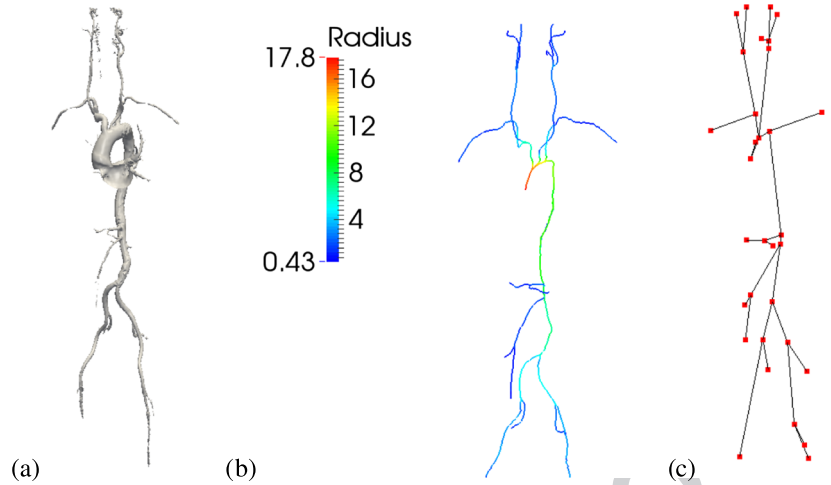
Data set	Case 1	Case 2
Resolution	512 × 512 × 248 voxels	512 × 512 × 211 voxels
Spacing	0.37 × 0.37 × 0.40 mm	0.46 × 0.46 × 0.48 mm
Aorta segmentation	5.80 sec	5.19 sec
Frangi filter	91.76 sec	73.94 sec

Table II. Data set resolution, CPU time of skeletonization, false twigs elimination and graph reconstruction stages, and total skeletal segments statistics for cardiac image Case 1 and segmented rabbit kidney.

Data set	Case 1	Rabbit kidney
Resolution	512 × 512 × 248 voxels	2000 × 1989 × 910 voxels
Distance map	0.20 sec	58.12 sec
Thinning	0.79 sec	526.98 sec
False twigs elimination	0.15 sec	16.61 sec
Graph construction	0.13 sec	12.27 sec
Number of skeletal segments	22 + 6 false twigs	4302 + 2142 false twigs

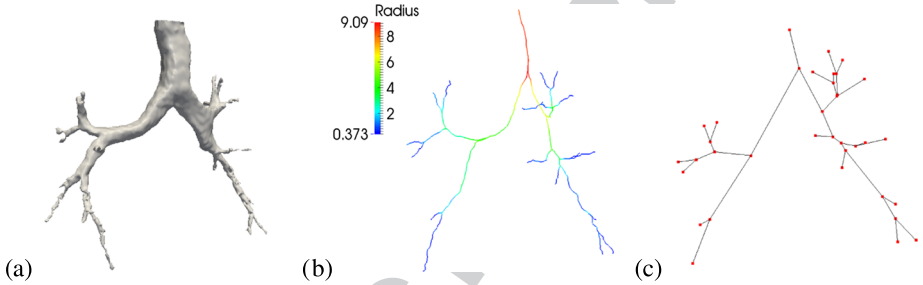
The DICOM images of micro-CT of vascular corrosion cast of rabbit kidney was kindly provided by J. Alastruey from the Department of Bioengineering, Imperial College London, UK. Papers [49, 50] present a method for rabbit kidney casting. The cast is μCT-imaged with high resolution 2000 × 1989 × 910 voxels. The data was split in three blocks, and for each block the Frangi filter was applied with scales $1 \leqslant s \leqslant 20$ voxels. Three segmented images were combined and the largest connected component was selected. We applied Frangi filter with parameters $\alpha = 0.4$, $\beta = 0.2$ and $c = 300$. The segmented image was used for skeletonization and false twigs elimination tests. The run times of different skeletonization stages for cardiac image Case 1 and segmented rabbit kidney are presented in Table II.

DOI: 10.1002/cnm



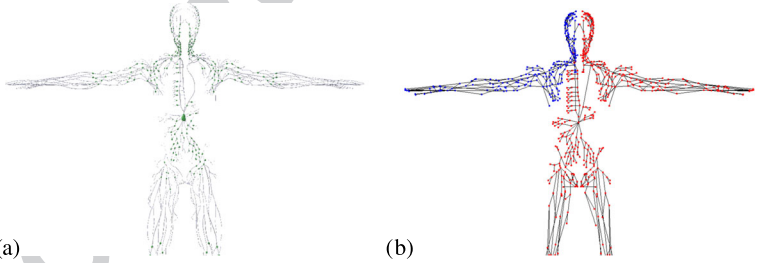
Color Online, B&W in Print

Figure 9. Reconstruction of large arteries: (a) segmentation, (b) extracted centerlines, and (c) 3D core graph. Q5



Color Online, B&W in Print

Figure 10. Reconstruction of human airways: (a) segmented trachea and bronchus,(b) extracted centerlines, and (c) 3D core graph.



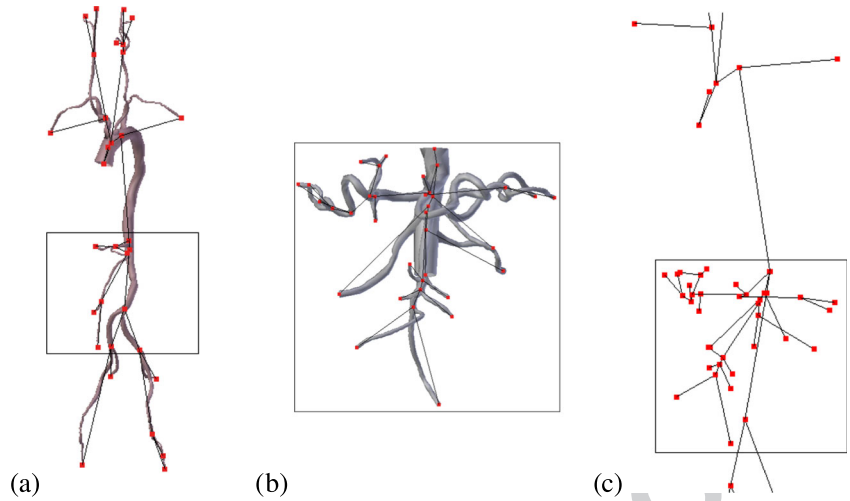
Color Online, B&W in Print

Figure 11. Two-component lymphatic structure extracted from realistic anatomy 3D model:(a) polygonal surface representation of lymphatic system and (b) 3D core graph.

The algorithm for centerlines-based graph reconstruction was applied to different preliminary segmented data with VMTK library methods: a number of CT datasets for arteries (Figures 9, 12, 13), MRI images of lungs region (Figure 10). Voxel size for all experiments is equal to 1mm × 1mm × 1mm. Figure 11 presents graph reconstruction of lymphatic system from the 3D-polygonal anatomically realistic model [45]. F9 F10 F11

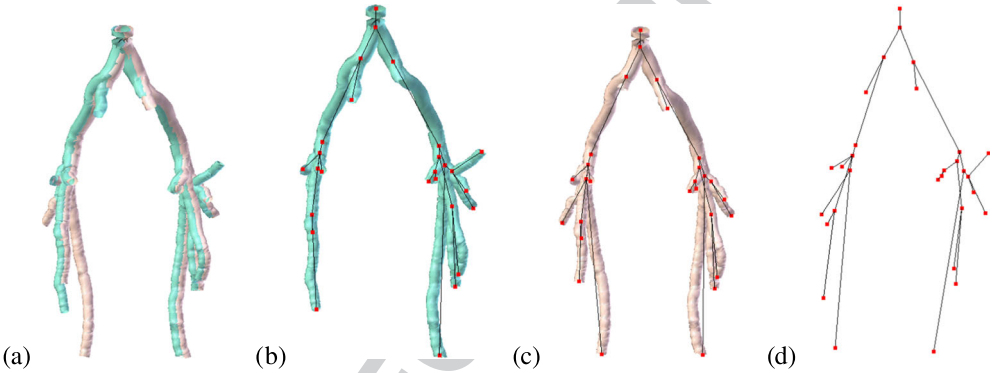
Figure 12 represents evaluation of the local-graph adaptation algorithm. A reference graph to be adapted was reconstructed from preliminary segmented main arteries for a patient (Figure 12a). Locally detailed graph was extracted from another patient and was considered as adaptation graph (Figure 12b). Figure 12c shows the result of the local-graph adaptation algorithm. F12

DOI: 10.1002/cnm



Color Online, B&W in Print

Figure 12. Performance of local adaptation method: (a) reference graph, (b) patient-specific graph reconstructed for local vascular domain, and (c) reference graph locally adapted.



Color Online, B&W in Print

Figure 13. Performance of graph matching method: (a) comparison of two different segmentation of the same vascular domain; (b) and (c) are reconstructed graphs for the first and the second segmentation; (d) merged graph constructed along with graph association method, respectively.

Graph matching method was tested on two artificially distorted segmentations of the same vascular network regions. These two graphs with different topology were automatically merged by the proposed method. The resulting merged graph is illustrated in Figure 13d.

7. CONCLUSIONS

The automated technology for personalized segmentation, skeletonization, and graph extraction was proposed in this paper. The following algorithms were presented in detail: fully automatic coronary artery segmentation, skeleton extraction from segmented image and automatic reconstruction of 1D network, automatic 1D network extraction from a set of centerlines with a common root, 1D network graphs postprocessing such as graph correction and local adaptation. Patient-specific data were used to demonstrate efficiency of 1D-network reconstruction algorithms. Constructed networks can be used in modeling of blood flows as well as other physiological processes in tubular structures.

ACKNOWLEDGEMENTS

The work was supported by the Russian Science Foundation (RSF) grant 14-31-00024. The authors acknowledge D.Burenchev and the staff of I.M.Sechenov First Moscow State Medical University and especially P.Kopylov, N.Gagarina, E.Fominykh, A.Dzyundzya for patient-specific data.

DOI: 10.1002/cnm

REFERENCES

1. Quarteroni A, Tuveri M, Veneziani A. Computational vascular fluid dynamics: problems, models and methods. *Computing and Visualization in Science* 2000; **2**:163–197.
2. Taylor CA, Hughes TJ, Zarins CK. Finite element modeling of blood flow in arteries. *Computer Methods in Applied Mechanics and Engineering* 1998; **158**:155–196.
3. Gerbeau JF, Vidrascu M, Frey P. Fluid–structure interaction in blood flows on geometries based on medical imaging. *Computers & Structures* 2005; **83**:155–165.
4. Sazonov I, Yeo SY, Bevan RLT, Xie X, van Loon R, Nithiarasu P. Modelling pipeline for subject-specific arterial blood flow – A review. *International Journal for Numerical Methods in Biomedical Engineering* 2011; **27**: 1868–1910.
5. Hughes TJ, Lubliner J. On the one-dimensional theory of blood flow in the larger vessels. *Mathematical Biosciences* 1973; **18**:161–170.
6. Formaggia L, Lamponi D, Quarteroni A. One-dimensional models for blood flow in arteries. *Journal of Engineering Mathematics* 2003; **47**:251–276.
7. Mynard JP, Nithiarasu P. A 1D arterial blood-flow model incorporating ventricular pressure, aortic valve and regional coronary flow using the locally conservative galerkin (lcg) method. *Communications in Numerical Methods in Engineering* 2008; **24**:367–417.
8. Alastruey J, Khir AW, Matthys KS, Segers P, Sherwin SJ, Verdonck PR, Parker KH, Peiró J. Pulse wave propagation in a model human arterial network: assessment of 1-D visco-elastic simulations against in vitro measurements. *Journal of Biomechanics* 2011; **44**:2250–2258.
9. Gamilov T, Ivanov Y, Kopylov P, Simakov S, Vassilevski Y, Sequeira A, Volpert V. Patient-specific haemodynamic modeling after occlusion treatment in leg. *Mathematical Modelling of Natural Phenomena* 2014; **9**:85–97.
10. Gamilov TM, Simakov SS. Modelling of coronary flow stimulation by enhanced external counterpulsation. Q6 *International Journal for Numerical Methods in Biomedical Engineering* 2015. Submitted.
11. Dobroserdova TK, Olshanskii MA. A finite element solver and energy stable coupling for 3D and 1D fluid models. *Computer Methods in Applied Mechanics and Engineering* 2013; **259**:166–176.
12. Holtzman-Gazit M, Kimmel R, Peled N, Goldsher D. Segmentation of thin structures in volumetric medical images. *IEEE Transactions on Image Processing* 2006; **15**:354–363.
13. Radaelli AG, Peiró J. On the segmentation of vascular geometries from medical images. *International Journal for Numerical Methods in Biomedical Engineering* 2010; **26**:3–34.
14. Yeo SY, Xie X, Sazonov I, Nithiarasu P. Segmentation of biomedical images using active contour model with robust image feature and shape prior. *International Journal for Numerical Methods in Biomedical Engineering* 2014; **30**:232–248.
15. Smistad E, Falch TL, Bozorgi M, Elster AC, Lindseth F. Medical image segmentation on GPUs – A comprehensive review. *Medical Image Analysis* 2015; **20**:1–18.
16. Waters SL, Alastruey J, Beard DA, Bovendeerd PH, Davies PF, Jayaraman G, Jensen OE, Lee J, Parker KH, Popel AS, *et al.* Theoretical models for coronary vascular biomechanics: progress & challenges. *Progress in Biophysics and Molecular Biology* 2011; **104**:49–76.
17. Grady L, Schwartz E. Isoperimetric graph partitioning for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence Machine Intelligence* 2006; **28**:469–475.
18. Grady L. Fast, quality, segmentation of large volumes – Isoperimetric distance trees. In *Computer Vision –ECCV 2006*, vol. 3953, Leonardis A, Bischof H, Pinz A (eds)., Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2006; 449–462.
19. Tek H, Gulsun MA, Laguitton S, Grady L, Lesage D, Funka-Lea G. Automatic coronary tree modeling. *The Midas* Q7 *Journal – 2008 MICCAI Workshop Grand Challenge Coronary Artery Tracking*, 2008. (Available from: http://hdl.handle.net/10380/1426).
20. Yang G, Kitslaar P, Frenay M, Broersen A, Boogers MJ, Bax JJ, Reiber JHC, Dijkstra J. Automatic centerline extraction of coronary arteries in coronary computed tomographic angiography. *International Journal of Cardiovascular Imaging* 2012; **28**:921–933.
21. Frangi A, Niessen W, Vincken K, Viergever M. Multiscale vessel enhancement filtering. In *Medical Image Computing and Computer-Assisted Interventation – MICCAI'98*, Wells W, Colchester A, Delp S (eds)., Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 1496, 1998; 130–137.
22. Pudney C. Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images. *Computer Vision and Image Understanding* 1998; **72**:404–413.
23. Blum H. A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*, Dunn WW (ed.). MIT Press: Cambridge, 1967; 362–380.
24. Reniers D, van Wijk J, Telea A. Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. *IEEE Transactions on Visualization and Computer Graphics* 2008; **14**:355–368.
25. Au OKC, Tai CL, Chu HK, Cohen-Or D, Lee TY. Skeleton extraction by mesh contraction. *ACM Transactions on Graphics* 2008; **27**(44):1–10.
26. Arcelli C, di Baja GS, Serino L. Distance-driven skeletonization in voxel images. *IEEE Transactions on Pattern Analysis and Machine Intelligence Machine Intelligence* 2011; **33**:709–720.
27. Tagliasacchi A, Alhashim I, Olson M, Zhang H. Mean curvature skeletons. *Computer Graphics Forum* 2012; **31**:1735–1744.

DOI: 10.1002/cnm

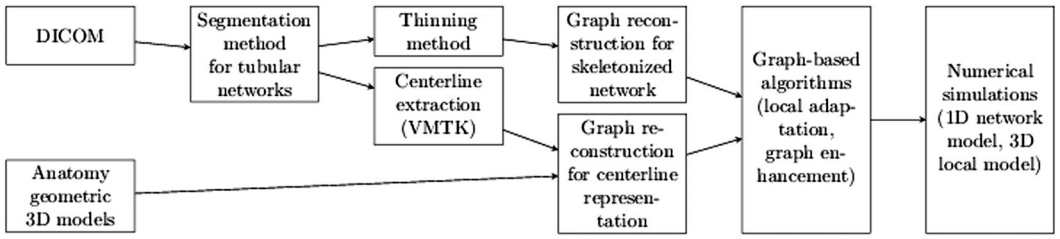
28. Stamatelos SK, Kim E, Pathak AP, Popel AS. A bioimage informatics based reconstruction of breast tumor microvasculature with computational blood flow predictions. *Microvascular Research* 2014; **91**:8–21.
29. VMTK – the vascular modeling toolkit. (Available from: http://www.vmtk.org/).
30. Wan M, Liang Z, Ke Q, Hong L, Bitter I, Kaufman A. Automatic centerline extraction for virtual colonoscopy. *IEEE Transactions on Medical Imaging* 2002; **21**:1450–1460.
31. Hassouna M, Farag A. Robust centerline extraction framework using level sets. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference* 2005; **1**:458–465.
32. Dey TK, Sun J. Defining and computing curve-skeletons with medial geodesic function. *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, Eurographics Association: Aire-la-Ville, Switzerland, Switzerland, 2006; 143–152.
33. Hesselink W, Roerdink J. Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence Machine Intelligence* 2008; **30**:2204–2217.
34. Sobiecki A, Yasan H, Jalba A, Telea A. Qualitative comparison of contraction-based curve skeletonization methods. In *Mathematical Morphology and Its Applications to Signal and Image Processing*, vol. 7883, Hendriks C, Borgefors G, Strand R (eds)., Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2013; 425–439.
35. Gülsün M, Tek H. Robust vessel tree modeling. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, vol. 5241, Metaxas D, Axel L, Fichtinger G, Székely G (eds)., Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 2008; 602–611.
36. Kumar R, Albregtsen F, Reimers M, Edwin B, Langø T, Elle O. Blood vessel segmentation and centerline tracking using local structure analysis. In *6th European Conference of the International Federation for Medical and Biological Engineering, IFMBE Proceedings*, Vol. 45, Lacković I, Vasic D (eds). Springer International Publishing, 2015.
37. Bai X, Latecki L, Liu Wy. Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence Machine Intelligence* 2007; **29**:449–462.
38. Palágyi K, Kuba A. Directional 3D thinning using 8 subiterations. In *Discrete Geometry for Computer Imagery*, vol. 1568, Bertrand G, Couprie M, Perroton L (eds)., Lecture Notes in Computer Science. Springer: Berlin Heidelberg, 1999; 325–336.
39. Pelillo M, Siddiqi K, Zucker S. Matching hierarchical structures using association graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence Machine Intelligence* 1999; **21**:1105–1120.
40. Florack LM, ter Haar Romeny BM, Koenderink JJ, Viergever MA. Scale and the differential structure of images. *Image and Vision Computing* 1992; **10**:376–388.
41. Lindeberg T. Edge detection and ridge detection with automatic scale selection. *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, 1996; 465–470.
42. Kong T, topology Rosenfeld A. Digital. Introduction and survey. *Computer Vision, Graphics, and Image Processing* 1989; **48**:357–393.
43. Bertrand G, Malandain G. A new characterization of three-dimensional simple points. *Pattern Recognition Letters* 1994; **15**:169–175.
44. Borgefors G. Distance transformations in arbitrary dimensions. *Computer Vision, Graphics, and Image Processing* 1984; **27**:321–345.
45. Plasticboy – Anatomy 3D models store. (Available from: http://www.plasticboy.co.uk/).
46. Boppana R, Halldórsson MM. Approximating maximum independent sets by excluding subgraphs. *BIT* 1992; **32**:180–196.
47. Metzen JH, Kröger T, Schenk A, Zidowitz S, Peitgen HO, Jiang X. Matching of anatomical tree structures for registration of medical images. *Image and Vision Computing* 2009; **27**:923–933.
48. Diedrich KT, Roberts JA, Schmidt RH, Parker DL. Comparing performance of centerline algorithms for quantitative assessment of brain vascular anatomy. *Anatomical Record* 2012; **295**:2179–2190.
49. Alastruey J, Nagel SR, Nier BA, Hunt AA, Weinberg PD, Peirò J. Modelling pulse wave propagation in the rabbit systemic circulation to assess the effects of altered nitric oxide synthesis. *Journal of Biomechanics* 2009; **42**: 2116–2123.
50. Alastruey J, Hunt AAE, Weinberg PD. Novel wave intensity analysis of arterial pulse wave propagation accounting for peripheral reflections. *International Journal for Numerical Methods in Biomedical Engineering* 2013; **30**: 249–279.

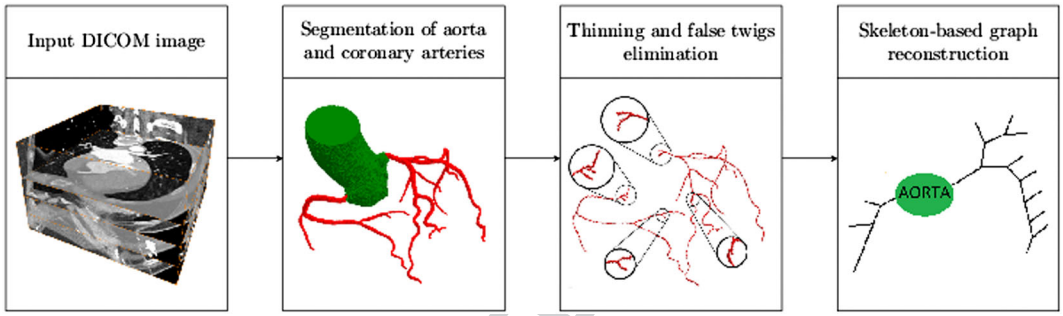
DOI: 10.1002/cnm

Research Article

Methods of graph network reconstruction in personalized medicine

A. Danilov, Yu. Ivanov, R. Pryamonosov and Yu. Vassilevski





The technology for personalized segmentation, skeletonization and graph extraction is proposed. The novelties of this paper are the following algorithms: fully automatic coronary artery segmentation, automatic reconstruction of 1D network using skeleton extraction from segmented image, automatic 1D network extraction from a set of centerlines with common root, 1D network graphs post-processing such as graph correction and local adaptation.

Author Query Form

Journal: International Journal for Numerical Methods in Biomedical Engineering

Article: cnm_2754

Dear Author,

During the copyediting of your paper, the following queries arose. Please respond to these by annotating your proof with the necessary changes/additions.

- If you intend to annotate your proof electronically, please refer to the E-annotation guidelines.
- If you intend to annotate your proof by means of hard-copy mark-up, please use the standard proofreading marks in annotating corrections. If manually writing corrections on your proof and returning it by fax, do not write too close to the edge of the paper. Please remember that illegible mark-ups may delay publication.

Whether you opt for hard-copy or electronic annotation of your proof, we recommend that you provide additional clarification of answers to queries by entering your answers on the query sheet, in addition to the text mark-up.

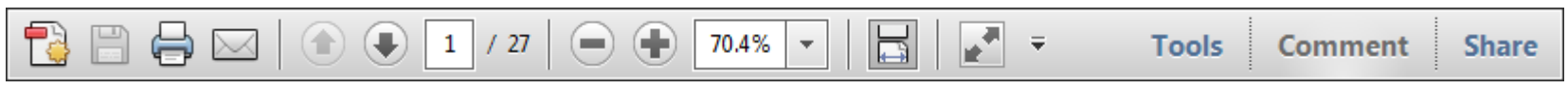
Query No.	Query	Remark
Q1	AUTHOR: One-dimensional. Is this the correct definition for 1D? Please check and confirm.	
Q2	AUTHOR: Three-dimensional. Is this the correct definition for 3D. Please check and confirm.	
Q3	AUTHOR: To maintain sequential order, sections, equations, and enunciations have been renumbered. Hence, their corresponding citations have also changed throughout the text. Please check and correct if necessary.	
Q4	AUTHOR: Figures 2, 5, 6 and 7 are recommended for color online and in print. Please check.	
Q5	AUTHOR: Figures 9 and 10 have been renumbered according to citation order. Please check.	
Q6	AUTHOR: If reference 10 has now been published online, please add relevant information. If this reference has now been published in print, please add relevant volume/issue/pag information.	
Q7	AUTHOR: Please provide accessed date for reference 19, 29 and 45.	
Q8	AUTHOR: Please provide city location for reference 36 and 41.	

WILEY

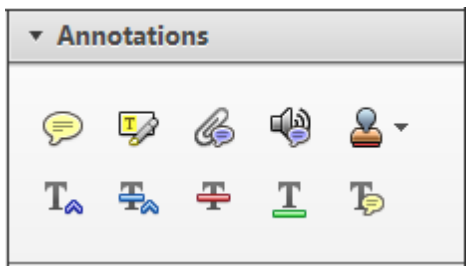
USING e-ANNOTATION TOOLS FOR ELECTRONIC PROOF CORRECTION

Required software to e-Annotate PDFs: Adobe Acrobat Professional or Adobe Reader (version 7.0 or above). (Note that this document uses screenshots from Adobe Reader X)
The latest version of Acrobat Reader can be downloaded for free at: http://get.adobe.com/uk/reader/

Once you have Acrobat Reader open on your computer, click on the Comment tab at the right of the toolbar:



This will open up a panel down the right side of the document. The majority of tools you will use for annotating your proof will be in the Annotations section, pictured opposite. We've picked out some of these tools below:

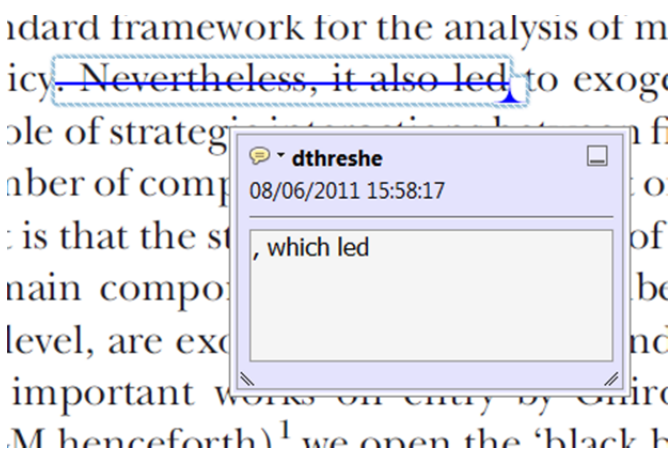


1. Replace (Ins) Tool – for replacing text.

Strikes a line through text and opens up a text box where replacement text can be entered.

How to use it

- Highlight a word or sentence.
- Click on the Replace (Ins) icon in the Annotations section.
- Type the replacement text into the blue box that appears.

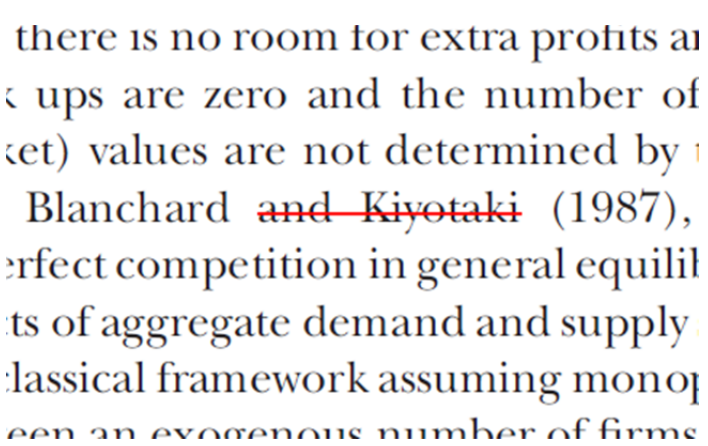


2. Strikethrough (Del) Tool – for deleting text.

Strikes a red line through text that is to be deleted.

How to use it

- Highlight a word or sentence.
- Click on the Strikethrough (Del) icon in the Annotations section.

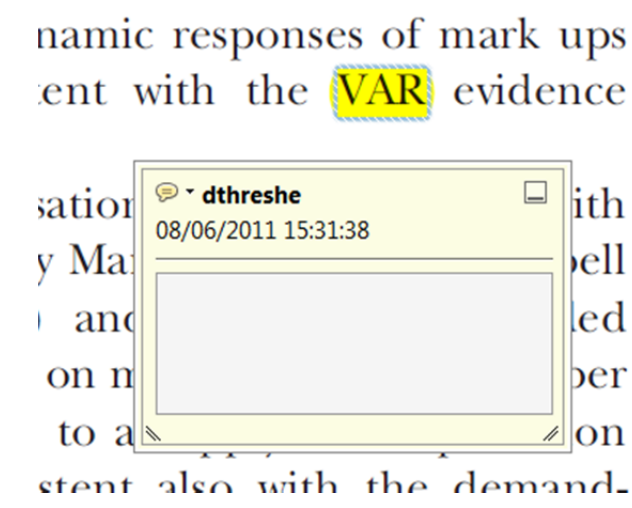


3. Add note to text Tool – for highlighting a section to be changed to bold or italic.

Highlights text in yellow and opens up a text box where comments can be entered.

How to use it

- Highlight the relevant section of text.
- Click on the Add note to text icon in the Annotations section.
- Type instruction on what should be changed regarding the text into the yellow box that appears.

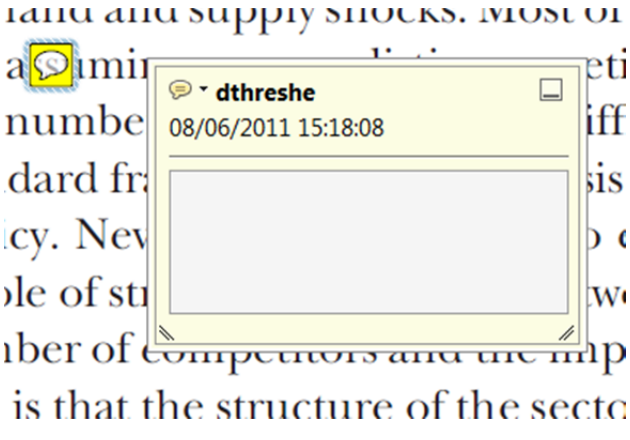


4. Add sticky note Tool – for making notes at specific points in the text.

Marks a point in the proof where a comment needs to be highlighted.

How to use it

- Click on the Add sticky note icon in the Annotations section.
- Click at the point in the proof where the comment should be inserted.
- Type the comment into the yellow box that appears.



WILEY

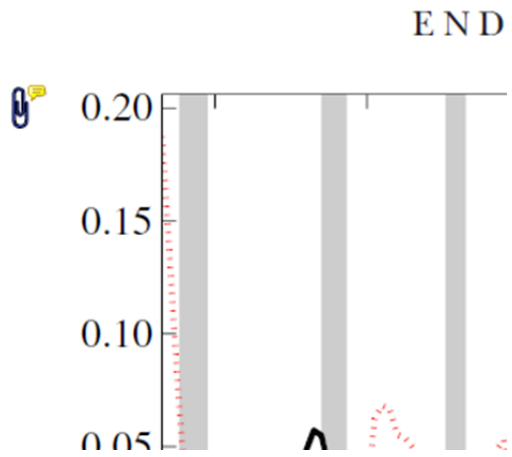
USING e-ANNOTATION TOOLS FOR ELECTRONIC PROOF CORRECTION

5. Attach File Tool – for inserting large amounts of text or replacement figures.

Inserts an icon linking to the attached file in the appropriate pace in the text.

How to use it

- Click on the Attach File icon in the Annotations section.
- Click on the proof to where you'd like the attached file to be linked.
- Select the file to be attached from your computer or network.
- Select the colour and type of icon that will appear in the proof. Click OK.

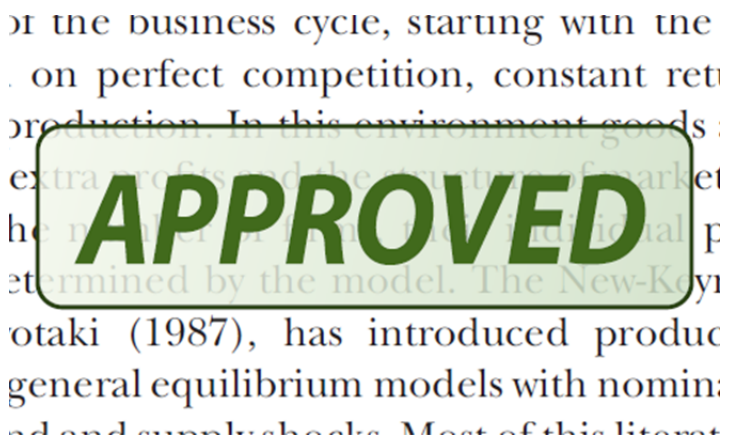


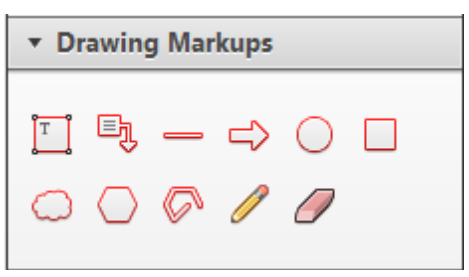
6. Add stamp Tool – for approving a proof if no corrections are required.

Inserts a selected stamp onto an appropriate place in the proof.

How to use it

- Click on the Add stamp icon in the Annotations section.
- Select the stamp you want to use. (The Approved stamp is usually available directly in the menu that appears).
- Click on the proof where you'd like the stamp to appear. (Where a proof is to be approved as it is, this would normally be on the first page).



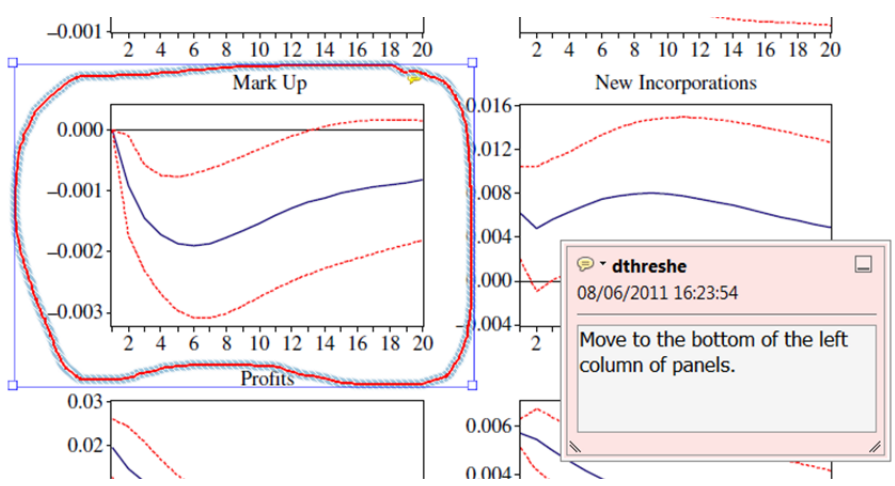


7. Drawing Markups Tools – for drawing shapes, lines and freeform annotations on proofs and commenting on these marks.

Allows shapes, lines and freeform annotations to be drawn on proofs and for comment to be made on these marks..

How to use it

- Click on one of the shapes in the Drawing Markups section.
- Click on the proof at the relevant point and draw the selected shape with the cursor.
- To add a comment to the drawn shape, move the cursor over the shape until an arrowhead appears.
- Double click on the shape and type any text in the red box that appears.



For further information on how to annotate proofs, click on the Help menu to reveal a list of further options: