Vis Comput (2009) 25: 627–635
DOI 10.1007/s00371-009-0325-5

ORIGINAL ARTICLE

Interactive skeletonization of intensity volumes

Sasakthi S. Abeysinghe · Tao Ju

Published online: 3 March 2009

Abstract We present an interactive approach for identifying skeletons (i.e. centerlines) in intensity volumes, such as those produced by biomedical imaging. While skeletons are very useful for a range of image analysis tasks, it is extremely difficult to obtain skeletons with correct connectivity and shape from noisy inputs using automatic skeletonization methods. In this paper we explore how easy-to-supply user inputs, such as simple mouse clicking and scribbling, can guide the creation of satisfactory skeletons. Our contributions include formulating the task of drawing 3D centerlines given 2D user inputs as a constrained optimization problem, solving this problem on a discrete graph using a shortest-path algorithm, building a graphical interface for interactive skeletonization and testing it on a range of biomedical data.

Keywords Interactive · Skeletonization · Intensity volumes

1 Introduction

Recent advances in biomedical imaging has allowed biological structures at various scales to be captured as digital volumes, such as MRI and CT. In these volumes, biological structures are encoded (implicitly) using *intensity* values stored at each voxel in a 3-dimensional array. The task of extracting a biological structure of interest from an intensity volume, and representing it in an appropriate form for analysis, is one of the core research problems in medical image processing.

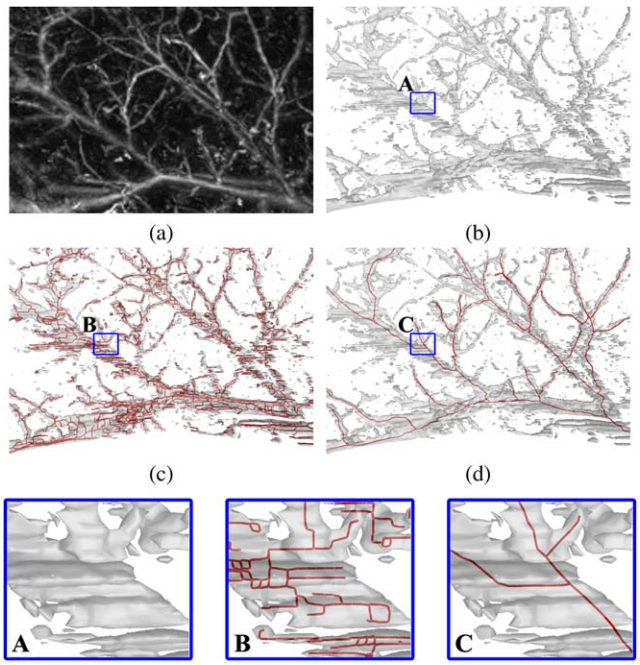


Fig. 1 A noise-abundant photo-acoustic scan of a subcutaneous vessel network (**a**), an iso-surface (**b**), the shape and topology preserving skeleton (*red lines*) generated by an automatic thinning method [14] from that iso-surface (**c**), and the skeleton created interactively using our tool (**d**) which does not depend on the iso-surface. The close-up views are compared at the bottom. The loops and extra/missing branches seen in B correspond to noise induced topology errors most prominent along the z-axis (not shown here)

A class of biological structures exhibit tubular shapes in 3D. Such structures exist at all scales, such as bronchial airways, blood vessels, nerve and muscle cells, and even proteins at near-atomic resolution. Knowledge of these struc-

S.S. Abeysinghe (✉) · T. Ju
Washington University in St. Louis, St. Louis, USA
e-mail: sasakthi.abeysinghe@wustl.edu

T. Ju
e-mail: taoju@cse.wustl.edu

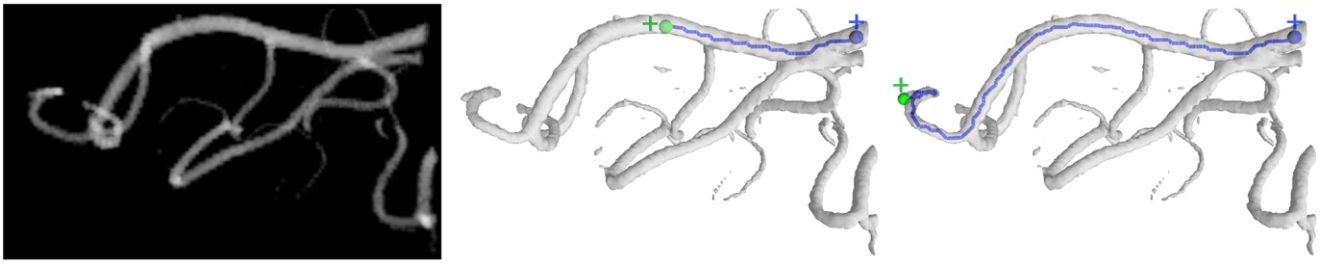


Fig. 2 Skeleton creation by end-pixel clicking: given an intensity volume (**a**) visualized by iso-surfaces at chosen iso-levels (**b**, **c**), the user can create a 3D skeleton curve by clicking two 2D end-pixels on the screen, shown in (**b**, **c**) as *green* and *blue crosses*

tures can often be drawn from an explicit geometric representation known as a *skeleton*, which is the collection of centerlines of these structures that depict their elongated shapes and connectivity. Skeletons enable automatic quantification of various structural properties, such as locations of tunnels and pathways in molecules [27], connectivity of protein backbones in virus complexes [13], cell lengths [20], blood vessel morphology [23] and cross-sectional areas [16].

Numerous automatic methods have been proposed in computer graphics and vision for skeleton extraction (see a brief review in Sect. 2). However, it is difficult to ensure the correctness of the resulting skeletons when the input data contains noise or ambiguity, which is very common in intensity volumes produced by biomedical imaging. An example is shown in Fig. 1, where the input volume (a) contains a significant amount of noise that leads to numerous errors in an automatically generated skeleton (c).

In this paper, we explore an interactive approach to skeletonization. Compared to automated algorithms, humans are superior at perception; allowing them to identify structures in images, even in the presence of noise or lack of sufficient resolution. Our goal is to develop a convenient interface that allows a user to "draw" a complex skeleton with minimal labor.

1.1 Overview

The main challenge in letting a user create the 3D skeleton is that the user has to determine the spatial location of skeleton points. To ease the creation process, we propose two interaction mechanisms.

Our first mechanism is inspired by the seminal work of Mortensen and Barrett [19]. They introduced "intelligent scissors" (also known as "live wire" [4]): an interactive way for drawing object boundary curves in a 2D image where the user only needs to click on the end-points of the curves. While we could let the user create a 3D skeleton curve by providing their 3D end-points, locating points in 3D is cumbersome from a user-interaction perspective. Instead, we allow users to create 3D skeleton curves by simply clicking on two *end-pixels* on the screen at any view of the volume. This is demonstrated in Fig. 2(b), where the intensity volume in (a) is viewed as an iso-surface, the two user-specified end-pixels are marked by blue and green crosses, and the 3D skeleton curve is shown in blue with their actual 3D end-points shown as spheres. As in "intelligent scissors", we allow a check-as-you-go style of skeleton creation: after the user clicks on a first end-pixel (e.g. blue cross in Fig. 2(b)), he or she can freely move the second end-pixel over the screen (e.g. green crosses in Fig. 2(b, c)) while watching the 3D skeleton curve updated on the fly.

We provide an additional interaction mode where the user can fine-tune the skeleton curve by *scribbling* on the screen. This is demonstrated in Fig. 3. In (a), the skeleton curve generated using only end-pixels go through an incorrect path. To correct the skeleton, the user scribbles along the desired path as he sees on the screen, as shown in green in (b). The skeleton curve is then updated, shown in (c), whose projection on the screen follows the scribbles.

Both interaction mechanisms allow certain amount of inaccuracy in user inputs. That is, the end-pixels or scribbles do not have to lie exactly at where the ends or the skeleton curve projects onto the screen. To achieve such interaction, we formulate the 3D skeleton curves intended by 2D user inputs (i.e. end-pixels and scribbles) as a solution to an optimization problem, where the optimization goals include the closeness of the projections of the skeleton end-points to the end-pixels and scribbles, the centeredness of the skeleton in the intensity volume, and the smoothness of the skeleton curve. The optimization problem is then solved on a discrete graph by a shortest-path algorithm. To allow on-the-fly computation of these shortest paths in real-world-size volumes (e.g. 256^3 voxels), we present means to construct a simplified graph with adaptive resolution.

1.2 Contributions

To the best of our knowledge, our work is the first that allows one to interactively create skeletons directly on a

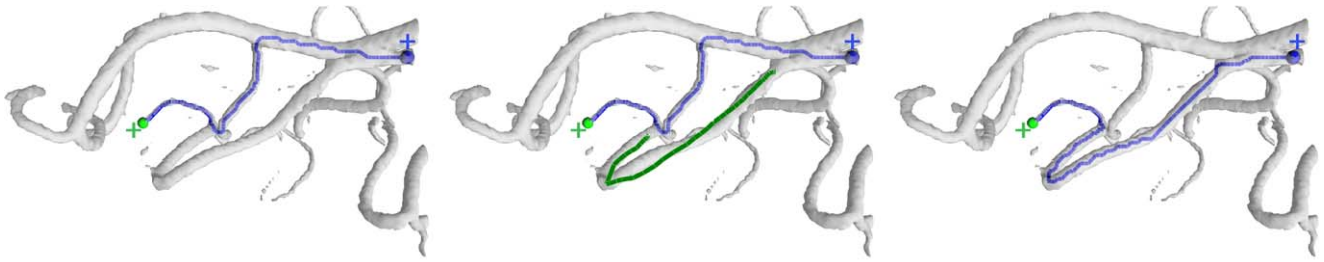


Fig. 3 Skeleton tuning by scribbling: given a skeleton curve created by end-pixels (**a**), the user can indicate the desirable path of the curve by a simple sketch on the screen, shown in *green* in (**b**), and the curve will be updated (**c**)

3D visualization of an intensity volume. While the use of minimal-cost paths between end-points have been explored previously for skeleton generation [7, 9] (see review in the next section), these and similar methods require users to locate the 3D end-points, which is cumbersome with 2D displays and input devices. Our work attempts to address the user-interaction challenges of 3D geometry creation, and we make the following technical contributions:

1. We introduce two types of user inputs: end-pixels and scribbles, and formulate 3D skeleton curves intended by these 2D inputs as solutions to a constrained energy minimization problem (Sects. 3.1, 3.2).
2. We solve the continuous minimization problem on a discrete graph structure using shortest-paths, and present a method to construct a graph with adaptive resolution for real-time interaction (Sect. 3.3).
3. We develop a graphical interface that allows easy creation and editing of 3D skeletons, which is particularly applicable to noise-abundant intensity volumes produced by biomedical imaging, such as in Fig. 1 (Sect. 4).

2 Previous work

We briefly review previous work on skeleton computation and on interactive techniques for extracting structures of interest from images and volumes.

Skeletonization There is a vast amount of literature on automatically computing skeleton curves. A majority of these methods are developed for solid objects, or *binary* images and volumes that have been segmented into object and background. We refer interested readers to the excellent book by Siddiqi et al. [24] and the survey by Cornea et al. [8]. The main limitation when applying these methods to intensity volumes in biomedical imaging is due to the difficulty in obtaining a good segmentation of the biological structure in the first place, which is in itself an interesting and open research problem. In particular, noise in the data can easily lead to inaccurate segmentations, which in turn leads to incorrect skeleton geometry and connectivity. For example, the spurious skeleton in Fig. 1(c) is generated from an inappropriate segmentation (b) using morphological thinning, which maintains the (noisy) topological structure of the segmentation. While methods that explicitly consider surface noise during skeletonization have been proposed [22], their results are still biased by the choice of the initial binary segmentation.

A number of methods have been proposed for automatic extraction of skeleton curves in an intensity volume without need for segmentation using gradient zero-crossing [10, 12], multiscale centerlines [15], and tracking or pruning using local structure tensors [1, 29]. While the resulting skeletons are more robust as they do not rely on the quality of segmentation, they can still be disrupted by the presence of noise [17].

Semi-automatic creation of skeleton curves as minimal-cost paths between user-specified end-points was pioneered by Cohen and co-authors [7, 9]. Recent studies [18, 28] show that for real-world biomedical data, this semi-automatic approach excels when compared to fully automatic approaches in terms of accuracy. However, existing minimal path based skeletonization methods require users to supply 3D end-points of the skeleton curves, typically on an appropriately-chosen 2D slice of the volume, which is cumbersome and time-consuming. In addition, the computations of the 3D minimal-paths in these methods do not reach interactive speeds, disallowing the check-as-you-go type of interaction that we aim to provide.

Interactive techniques for structure delineation Interactive techniques for extracting structures of interest in 2D images or 3D volumes have been abundant in recent literature. The majority of these methods focus on delineating the boundary curve (2D) or surface (3D) of the structure, such as using region-growing with interactive initialization [26, 30], interactive graph-cut [3, 5], interactive boundary drawing [11, 19, 21], to name a few. We see our work making a novel addition to this literature that provides a convenient 2D interface for drawing 3D skeletons. In addition, the resulting

skeletons produced by our approach can be used as initializers or seeds in subsequent region-growing or graph-cut segmentations of the structures themselves.

3 Method

Directly "drawing" 3D geometry is a challenging user interaction problem, particularly due to the difficulty in providing depth information using 2D input devices. We combine convenient screen-space user inputs with efficient algorithms to address this challenge when creating 3D skeletons. The user inputs serve as indicators of where the user wants the skeleton to be, and the algorithm is responsible for computing the actual skeleton that fulfills the users' intentions.

3.1 User inputs

The skeleton is created by piecing individual skeleton curves. To create one skeleton curve, we consider the following two types of user inputs:

End-pixels The user can indicate the two ends of a 3D skeleton curve by clicking on their projected pixels on the screen (e.g. crosses in Fig. 2(b, c)). These two end-pixels can be specified on different views. Formally, we denote p, q the two end-pixels (in screen coordinates), and M_p, M_q their respective viewing transformation matrices so that $M_p \cdot x$ projects a 3D point x onto the same view plane on which p lies.

Scribbles The user can indicate the desired path of the skeleton curve by scribbling along the projection of the path on the screen (as shown in Fig. 3(b)). For iterative improvement, we allow multiple scribbles to be specified in different views. Formally, we denote $\{r_i\}$ the set of all scribble points (in screen coordinates) and $\{M_{r_i}\}$ their corresponding viewing transformation matrices.

3.2 Defining skeletons

Given a pair of user-provided end-points and possibly some scribbles in screen-space, we can characterize the "intended" 3D skeleton curve as follows. First, the curve should lie close to the centerlines in the intensity volume. Second, the projections of the two curve ends should lie close to the user-provided end-pixels on their respective view planes. Third, the projection of the curve itself should lie close to the user-provided scribbles, if any, on their view planes.

We can define the curve with the above characteristics as the solution to a constrained energy minimization problem. Given any continuous spatial curve $S(t)$ parameterized by $t \in [0, 1]$, we define an energy of the form

$$\int_0^1 E\big(S(t), S'(t)\big)\, dt \tag{1}$$

where $S'(t)$ denotes the (unit) tangent direction of the curve at t, and E (to be formulated next) is a cost function that penalizes points that do not lie close to the centerlines in the intensity volume or whose projections deviate from the scribbles. The desired skeleton curve then becomes the one that minimizes (1) subject to constraints:

$$\big\| M_p \cdot S(0) - p \big\| \le \epsilon \quad \text{and} \quad \big\| M_q \cdot S(1) - q \big\| \le \epsilon$$

where ϵ is a user-specified tolerance (in terms of screen pixels) around the end-pixels.

Now we present our formulation of the cost function E, which has the following form balanced by a user-defined weight α:

$$E(x, \mathbf{v}) = \big(C(x) + \alpha D(x, \mathbf{v})\big) R(x) \tag{2}$$

where

- $C(x)$ evaluates the centeredness of a spatial point x. Assuming that the "centers" in the intensity volume are located at locally high intensity regions (see discussion in Sect. 6), we have

  $$C(x) = \frac{I_{\max}(x) - I(x)}{I_{\max}(x) - I_{\min}(x)}$$

  where $I(x)$ evaluates the intensity at point x, and $I_{\max}(x)$, $I_{\min}(x)$ are the maximum and minimum intensities in the local neighborhood centered around x within a user-specified radius σ. Hence $C(x) \in [0, 1]$ and is smaller if x lies closer to the local intensity maximum.
- $D(x, \mathbf{v})$ evaluates how the direction $\mathbf{v}$ is aligned with the local directionality in the intensity volume, and plays the role of ensuring the smoothness of the skeleton curve (a similar measure was adopted in [4] to ensure the smoothness of the boundary curve). Our formulation of D is from the method of Abeysinghe et al. [1], which computes the directionality in an intensity volume by examining the eigensystem of a local structure tensor. Let the eigenvectors and eigenvalues of the structure tensor at x within a neighborhood radius of σ be $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ and u_1, u_2, u_3, so that $u_1 \ge u_2 \ge u_3$, and let $v_i = \mathbf{v} \cdot \mathbf{e}_i$ for $i = 1, 2, 3$ (we refer readers to details in computing the structure tensor to literature such as [25]). The function D is written as:

  $$D(x, \mathbf{v}) = 1 - \frac{\sqrt{u_2^2 u_3^2 v_1^2 + u_1^2 u_3^2 v_2^2 + u_1^2 u_2^2 v_3^2}}{u_1 u_2}$$

  Like C, D returns a value between 0 and 1, and is smaller if $\mathbf{v}$ is better aligned with the local directionality.

– $R(x)$ evaluates how much the projection of x deviates from the shape scribbles, if there are any. The function has an inverse Gaussian-like shape:

$$R(x) = 1 + \min_i \left(1 - e^{\frac{-\|M_{r_i} \cdot x - r_i\|^2}{2h^2}}\right) H \tag{3}$$

where H is a large constant, and h is the user-specified kernel size that indicates how fast $R(x)$ increases as the distance $\|M_{r_i} \cdot x - r_i\|$ grows. The addition of 1 is to avoid a zero cost for points that directly project onto the scribbles due to the multiplication in (3). If no scribbles are provided, we let $R(x) = 1$.

3.3 Computing skeletons

To efficiently solve the energy minimization problem defined above, we shall present a discrete version of the continuous problem, whose global minimum can be easily found using graph algorithms.

3.3.1 Discrete formulation and solution

Let us consider a discrete version of the minimization problem where the spatial curves $S(t)$ are restricted to paths on a spatial graph with nodes $\{x_i\}$ and edges $\{x_i, x_j\}$. Given a path consisting of a sequence of edge-connected nodes $\{x_1, \ldots, x_k\}$, we can re-write the continuous energy in equation 1 as a discrete sum:

$$\sum_{i=1}^{k-1} E_e(x_i, x_{i+1}) \tag{4}$$

Here, E_e evaluates the energy along each edge in the path, which we approximate by discrete sampling:

$$E_e(x, y) = \|y - x\| \frac{\sum_{i=1}^{m} E(s_i, y - x)}{m} \tag{5}$$

where $\{s_1, \ldots, s_m\}$ are points sampled along the edge $\{x, y\}$ (including x and y) at a uniform interval no greater than a user-given maximum sampling distance. Now, our goal is to find the *path* that minimizes (4) subject to the constraints:

$$\|M_p \cdot x_1 - p\| \leq \epsilon \quad \text{and} \quad \|M_q \cdot x_k - q\| \leq \epsilon \tag{6}$$

The solution to the discrete minimization problem can be easily found as follows. We weight each graph edge $\{x_i, x_j\}$ by $E_e(x_i, x_j)$, and compute the shortest paths between all pairs of nodes that satisfy the end-pixel constraints above. The shortest among these paths is the globally minimal solution.

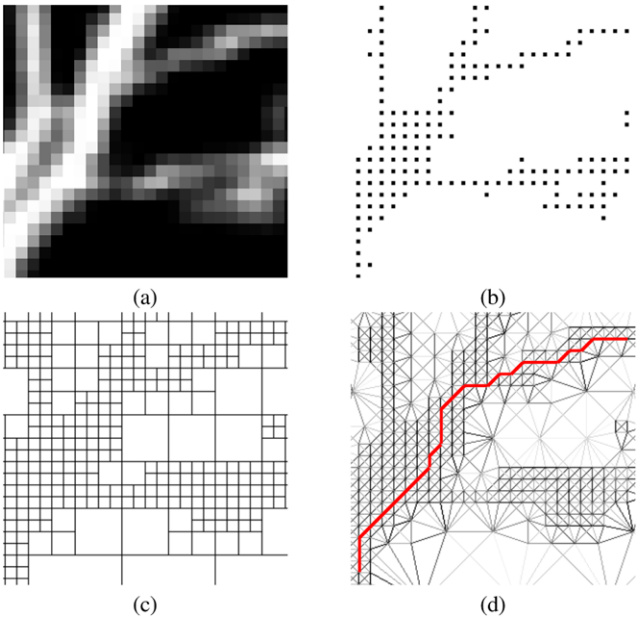


Fig. 4 A 2D image (**a**), its core pixels (**b**), the quad-tree representation of the image (**c**), and the resulting graph (**d**) with edges weighted (*darker color* indicates smaller weight) and a shortest path (*red*) between two nodes

3.3.2 Adaptive graph construction

A naïve way to construct the spatial graph from an intensity volume is to represent each voxel in the volume as a node and every pair of neighboring voxels as an edge. In this graph, finding a minimal-cost skeleton curve would invoke $O(n^2)$ number of shortest-path computations where n is the size of the volume, since there will be $\epsilon^2 n$ nodes to consider at either end of the curve.

We now present a way to construct a simplified graph that dramatically reduces the number of nodes that need to be considered during skeleton curve computation. The key observation is that, since we are looking for centerlines, we do not expect the skeleton curves to visit regions in the volume that are not likely to be centerlines. Hence we can use much fewer nodes in those regions without affecting the quality of the skeleton.

To achieve this effect, we will first select a subset of *core voxels* in the volume that are likely to lie along centerlines. While we could pick those voxels with a high centeredness function value $C(x)$, which is equivalent to local gradient based thresholding, such thresholding is likely to miss centerlines with low intensity gradients. Instead, we apply an existing automatic skeletonization approach that respects all intensity levels and gradients [1], and treat voxels in the resulting skeleton as core voxels. More specifically [1] obtains the skeletons in an intensity volume in two separate steps, thinning and pruning. To ensure that the core voxels maximally capture possible skeleton curves in the volume, we

only call the first step of the algorithm to compute an unpruned skeleton.

Given the core voxels, we create an octree representation of the volume where core voxels occupy the leaf cells at the finest level, and the tree is maximally collapsed elsewhere. Figure 4 is a 2D example, where the core pixels in the image (a) are shown in (b), and the resulting quad-tree representation of the image is shown in (c). The graph is then constructed by treating the center of each leaf cell (at all levels of the tree) as a node, and connecting every two neighboring cells by an edge, as shown in (d). The weight E_e for each edge is computed by (5) where the maximum sampling distance along the edge is chosen as the maximum distance between two adjacent voxels in the original volume (i.e. $\sqrt{3}$ in 3D when 26-adjacency is used).

Observe from Fig. 4(d) that the graph has an adaptive structure that is refined along possible skeleton curves. In addition, the weighting of the graph edges (darker shade represents lower weights) correspond well to the likelihood of the edges being part of a centerline. As an example, a shortest path is plotted between two nodes in the tree, which aligns well with the centerline.

4 User interface

The interaction mechanisms and algorithms described above were incorporated to an interactive tool called *Gorgon* [2] which we developed. Some example interactions in *Gorgon* are already shown in Figs. 2 and 3. Below we detail the usage of the tool and some implementation aspects.

4.1 Usage

As shown in Figs. 2 and 3, the intensity volume is displayed as a transparent iso-surface where the iso-level can be interactively adjusted to gain a better understanding of the overall shape. The user can perform rotations, translations and scaling using simple mouse clicks and drags.

To create a single skeleton curve, the user would start by clicking on an end-pixel on a selected view (e.g. blue crosses in Fig. 2). The user can then move the second end-pixel on the screen (e.g. green crosses in Fig. 2) while watching the 3D skeleton curve between the two end-pixels get updated on the fly. The user can also provide scribbles (colored green in Fig. 3(b)), and the curve will be updated. The user can repeat these two actions (e.g. moving the second end-pixel and more scribbling) on this and other views of the volume to adjust the skeleton curve while keeping the first end-pixel fixed. The skeleton curve is colored blue during this time, and will turn to red when the user is satisfied with it.

When scribbling, the user is free to draw any curve on the screen to indicate the intended skeleton path. This curve can even be self-intersecting on the screen, as demonstrated in Fig. 5. Such freedom makes it easy to create winding skeleton curves.

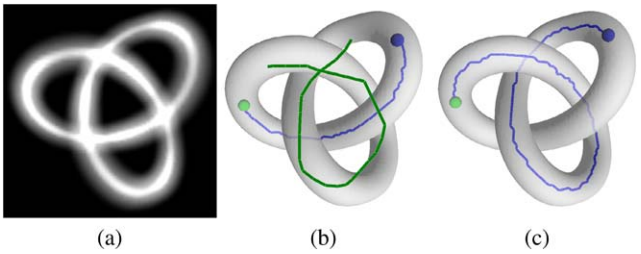


Fig. 5 A synthetic volume whose iso-surface forms a knot (**a**), the initial skeleton curve (*blue*) and a user-provided scribble (*green*) that indicates a different skeleton path (**b**) and the updated skeleton curve (**c**)

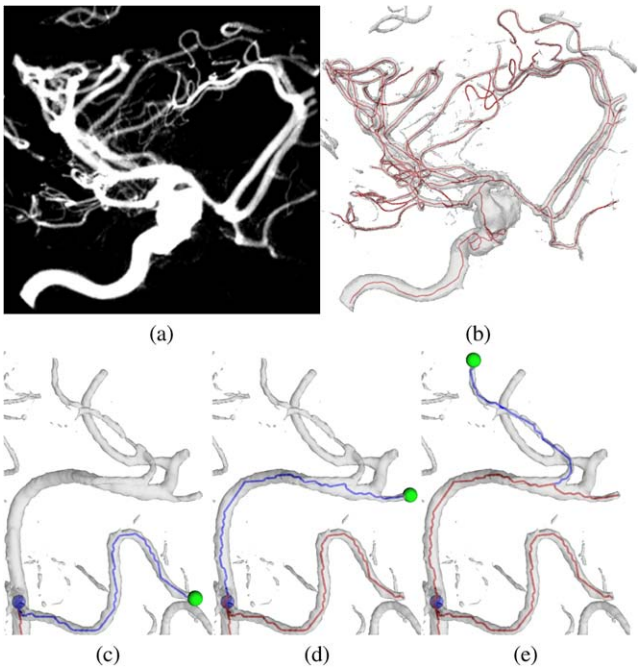


Fig. 6 A rotational C-arm x-ray scan of a blood vessel network in the human brain which contains an aneurism (**a**) and the final skeleton (**b**), intermediate steps in drawing the skeleton using the tree-mode (**c**, **d**, **e**)

Our tool provides convenient features to create a complex skeleton from individual skeleton curves. For example, it allows you to create a long skeleton curve from shorter, consecutive pieces by setting the terminal end of a previously drawn curve to be the starting point of the next curve (which we call the poly-line mode). For drawing tree-like structures such as blood vessels, the tool provides a second mode where, after the user creates a first curve, it sets the starting point of that curve to be the starting point of subsequent curves. The tree-mode is demonstrated in Fig. 6(c, d, e) when drawing a complex vessel tree. Some additional features include snapping ends of the new skeleton curves to

graph nodes on the current skeleton. We refer readers to the accompanying video for these features in action.

4.2 Implementation

When the user first loads a volume, a one-time pre-computation is performed to construct the graph as described in Sect. 3.3.2. After the user provides an end-pixel, we identify the graph nodes satisfying constraints in (6) by intersecting the volume with a *view cylinder* whose radius is ϵ and whose axis is the ray shot from the viewpoint to the end-pixel. This intersection can be efficiently computed using the octree structure from which the graph is built. To speed up the shortest path computation while user moves around the second end-pixel, we pre-compute and store the minimum spanning tree of every node within the view cylinder of the first end-pixel. This reduces subsequent computation of the shortest path from any of these nodes to a new node to linear time, allowing the skeleton curve to be found at an interactive rate. When scribbles are drawn, we re-weight the graph as the scribbling function $R(x)$ is updated in (2), and the spanning trees are re-computed. The default parameters (which were used in all our examples) are $\epsilon = 10$, $\alpha = 1$, $\sigma = 4$, $H = 1000$ and $h = 4$.

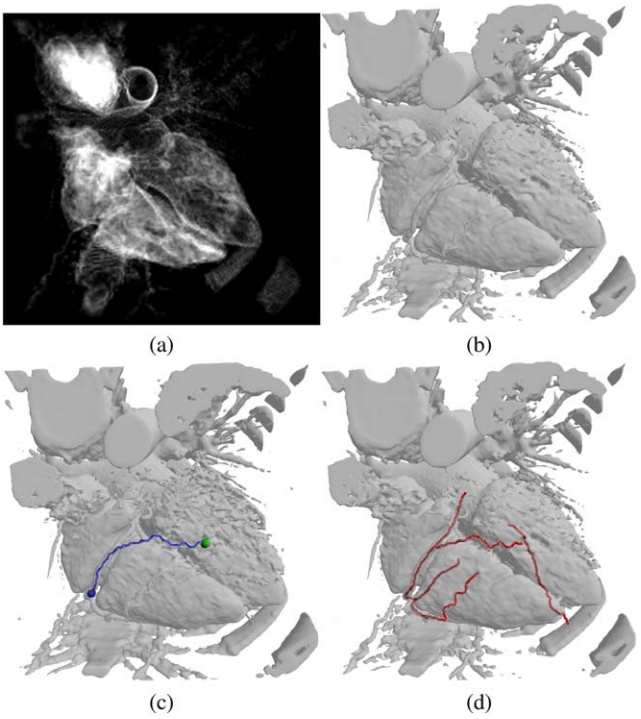


Fig. 7 A CT scan of the heart and surrounding area (**a**), an iso-contour where the coronary arteries are visible (**b**), an intermediate step in our method (**c**) and the final skeleton where the left and right coronary arteries have been skeletonized (**d**)

5 Results

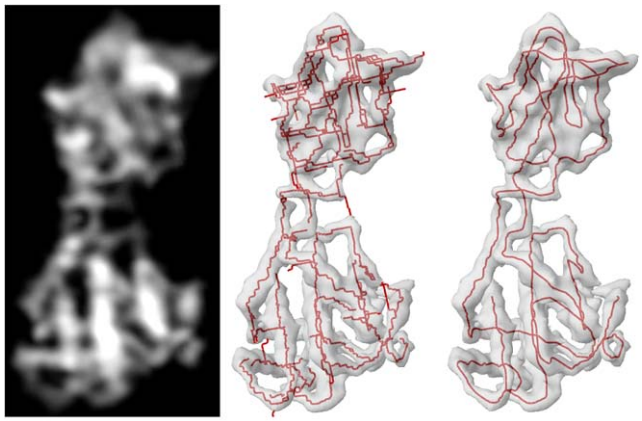
We apply our interface to extract structures from volumes created using various imaging modalities.

Figure 6 shows a rotational C-arm x-ray scan of a blood vessel network in the human brain. Note that the tool allows the user to draw skeletons of vessels at varying intensity levels while ensuring the correct connectivity, which is difficult for automatic algorithms. Figure 7 shows our technique being used to extract the centerlines of the left and right coronary arteries in a CT scan. Note that the interactive nature of our method makes it very easy for the user to capture features of interest (the coronary arteries in this case) while ignoring the rest, which is in itself a very hard problem for fully automated techniques.

Photo-acoustic imaging is an increasingly popular 3D imaging technique used to capture subcutaneous blood vessels [31]. As seen in Fig. 1 this technique generates reasonably accurate scans when in the x–y plane (a), but has a large amount of noise along the z-axis due to the temporal movements of the vasculature while the scan is in progress. Such anisotropic noise leads to incorrect skeleton connectivity when using automatic skeletonization techniques (c) and makes our interactive tool a more favorable choice (d).

As a last example, we demonstrate an application of our tool in protein modeling. Electron cryo-microscopy (cryo-EM) is a dominant protein imaging technique for medium to large sized macromolecular complexes, such as viruses [6]. An example of a cryo-EM volume of a single protein is shown in Fig. 8(a). Due to the lack of sufficient resolution, identifying atomic structures of a protein, such as its backbone (a string of amino acids), from cryo-EM volumes is difficult. An approach being used in the biology community [13] is to interactively place amino acid units along the centerlines in the cryo-EM volume. Previously, such centerlines were created by automatically skeletonizing the cryo-EM volume, which could contain numerous connectivity errors, such as loops and breaks (b). Using our tool, the biologist can create a much more accurate centerline (as a continuous, loop-free path) based on his knowledge of the protein backbone structure (c), which is important for obtaining a correct placement of amino acid units in the next step.

Performance The size of each data set used in our experiments and the performance of our tool on a PC with a 3 GHz Pentium-D CPU and 4 GB memory are reported in Table 1. Note that while the initial graph construction time may take up to 158 seconds, it does not affect subsequent interactions as it is a one-time process when the data is loaded. In our experience, the most frequent computation is updating the skeleton curve as the user moves the second end-pixel around while the first end-pixel is fixed (as in Fig. 2(b, c)), which is performed at interactive rate in all examples.



(a) (b) (c)

Fig. 8 A cryo-EM scan of a protein in the Blue Tongue Virus (BTV) (**a**), the skeleton created automatically using a segmentation-free skeletonization technique [1] (**b**) and using our tool (**c**)

Table 1 Running time for initial graph construction (a), computing minimum spanning trees after placing the first end-pixel (b), computing the skeleton curves as user moves around the second end-pixel (c), updating curve after user scribbles (d) and the total interaction time we used to create each skeleton (e) (excluding time (a))

Name	Size x:y:z	Time (a)	Time (b)	Time (c)	Time (d)	Time (e)
Figure 1	200:120:120	39.0 s	1.5 s	0.001 s	1.7 s	10 min
Figure 6	256:256:256	66.6 s	3.7 s	0.001 s	3.4 s	18 min
Figure 7	128:128:128	157.3 s	5.1 s	0.001 s	13.3 s	4 min
Figure 8	128:128:128	21.8 s	0.3 s	0.001 s	0.5 s	3 min

6 Conclusion and discussion

In this paper we proposed an innovative approach for skeletonizing intensity volumes. Our method does not require any explicit segmentation of the volume, is robust under the presence of noise, and is easy and intuitive to use. We tested our technique on real-world medical data sets captured using different imaging techniques to demonstrate its potential in different application domains.

While we have mainly demonstrated the use of the tool in creating skeletons from scratch, we are exploring how to combine the tool with existing skeletonization algorithms. One immediate approach is to import skeletons generated by these algorithms into our tool, which we can allow a user to inspect and edit using our interaction mechanisms.

As discussed in Sect. 3.2, the energy functions used in our approach assume that the volume's centerlines exhibit local intensity maxima. This condition may not always hold for certain imaging modalities, and even for different structures within a same image. To address this limitation, we will explore suitable "centeredness" measures for specific imaging modalities and material properties of a structure.

Another natural extension of our method is to add the ability to draw not only skeletal curves, but also skeletal surfaces, in intensity volumes. While skeletal curves are good representations for tubular objects, skeletal surfaces are useful descriptors of plate-like objects, such as the outer shell of bones, the β-sheets of proteins, etc. To be able to formulate and solve the constrained optimization problem for the surface case, a possible approach is to approximate a minimal-energy surface by stacking minimal-energy curves, which was adopted in interactive boundary-segmentation methods such as [21].

Acknowledgements We would like to thank Hao Zhang of the University of Wisconsin, Philips Research, the Erasmus medical center and Matthew Baker of the Baylor college of medicine for the data sets used in Figs. 1, 6, 7, 8 respectively. This work is supported in part by NSF grant IIS-0705538.

References

1. Abeysinghe, S.S., Baker, M., Chiu, W., Ju, T.: Segmentation-free skeletonization of grayscale volumes for shape understanding. In: Shape Modeling and Applications. IEEE International Conference on SMI 2008, pp. 63–71 (2008)
2. Abeysinghe, S.S., Coleman, R., Marsh, M., Ruths, T., Baker, M., Chiu, W., Ju, T.: Gorgon—an interactive molecular modeling system. http://www.cs.wustl.edu/~ssa1/gorgon/
3. Armstrong, C.J., Price, B.L., Barrett, W.A.: Interactive segmentation of image volumes with live surface. Comput. Graph. **31**(2), 212–229 (2007)
4. Barrett, W.A., Mortensen, E.N.: Interactive live-wire boundary extraction. Med. Image Anal. **1**(4), 331–341 (1997)
5. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient n–d image segmentation. Int. J. Comput. Vis. **70**(2), 109–131 (2006)
6. Chiu, W., Baker, M., Jiang, W., Dougherty, M., Schmid, M.: Electron cryomicroscopy of biological machines at subnanometer resolution. Structure (Camb.) **13**, 363–372 (2005)
7. Cohen, L.D., Kimmel, R.: Global minimum for active contour models: A minimal path approach. Int. J. Comput. Vis. **24**(1), 57–78 (1997)
8. Cornea, N.D., Min, P.: Curve-skeleton properties, applications, and algorithms. IEEE Trans. Vis. Comput. Graph. **13**(3), 530–548 (2007)
9. Deschamps, T., Cohen, L.D.: Minimal paths in 3d images and application to virtual endoscopy. In: ECCV '00: Proceedings of the 6th European Conference on Computer Vision—Part II, pp. 543–557. Springer, London (2000)
10. Eberly, D., Gardner, R., Morse, B., Pizer, S., Scharlach, C.: Ridges for image analysis. J. Math. Imaging Vis. **4**(4), 353–373 (1994)
11. Falcao, A., Udupa, J., Miyazawa, F.: An ultra-fast user-steered image segmentation paradigm: live wire on the fly. IEEE Trans. Med. Imaging **19**(1), 55–62 (2000)
12. Furst, J.D., Pizer, S.M., Eberly, D.H.: Marching cores: A method for extracting cores from 3d medical images. In: MMBIA '96: Proceedings of the 1996 Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA '96), p. 124. IEEE Computer Society, Washington (1996)
13. Jiang, W., Baker, M.L., Jakana, J., Weigele, P.R., King, J., Chiu, W.: Backbone structure of the infectious 15 virus capsid revealed by electron cryomicroscopy. Nature **451**, 1130–1134 (2008)
14. Ju, T., Baker, M., Chiu, W.: Computing a family of skeletons of volumetric models for shape description. Comput. Aided Des. **39**(5), 352–360 (2007)

15. Krissian, K., Malandain, G., Ayache, N., Vaillant, R., Trousset, Y.: Model-based detection of tubular structures in 3d images. Comput. Vis. Image Underst. **80**(2), 130–171 (2000)
16. Krissian, K., Vaillant, R., Trousset, Y., Malandain, G., Ayache., N.: Automatic and accurate measurement of a cross-sectional area of vessels in 3d x-ray angiography images. In: Proc. of SPIE 2000 (2000)
17. Krissian, K., Kikinis, R., Westin, C.F.: Algorithms for extracting vessel centerlines. Tech. Rep. 0003, Department of Radiology, Brigham and Women's Hospital, Harvard Medical School, Laboratory of Mathematics in Imaging (2004)
18. Metz, C.T., Schaap, M., van Walsum, T., van der Giessen, A., Weustink, A., Mollet, N., Krestin, G., Niessen, W.: 2008 MICCAI workshop: 3d segmentation in the clinic: A grand challenge II—coronary artery tracking. http://hdl.handle.net/10380/1399
19. Mortensen, E.N., Barrett, W.A.: Intelligent scissors for image composition. In: SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, pp. 191–198. ACM, New York (1995)
20. Nathan, A., Lee, S.L., Marquez, J., Ju, T., Pryse, K., Elson, E., Genin, G.M.: Active and passive responses of myofibroblasts in response to mechanical stretching in 3d culture. In: Proceedings of the ASME 2008 Summer Bioengineering Conference (SBC2008) (2008)
21. Poon, M., Hamarneh, G., Abugharbieh, R.: Efficient interactive 3d livewire segmentation of complex objects with arbitrary topology. Comput. Med. Imaging Graph. **32**(8), 639–650 (2008)
22. Reniers, D., van Wijk, J., Telea, A.: Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. IEEE Trans. Vis. Comput. Graph. **14**(2), 355–368 (2008)
23. Selle, D., Preim, B., Schenk, A., Peitgen, H.: Analysis of vasculature for liver surgical planning. IEEE Trans. Med. Imaging **21**(11), 1344–1357 (2002)
24. Siddiqi, K., Pizer, S. (eds.): Medial Representations: Mathematics, Algorithms and Applications. Springer, Berlin (2008)
25. Weickert, J.: Anisotropic Diffusion in Image Processing. Teubner, Leipzig (1998)
26. Whitaker, R.T., Breen, D.E., Museth, K., Soni, N.: A framework for level set segmentation of volume datasets. In: Volume Graphics (2001)
27. Wong, C.J., Rice, R.L., Baker, N.A., Ju, T., Lohman, T.M.: Probing 3′-ssdna loop formation in E. coli recbcd/recbc-dna complexes using non-natural DNA: A model for "chi" recognition complexes. J. Mol. Biol. **362**(1), 26–43 (2006)
28. Yim, P., Choyke, P., Summers, R.: Gray-scale skeletonization of small vessels in magnetic resonance angiography. IEEE Trans. Med. Imaging **19**(6), 568–576 (2000)
29. Yu, Z., Bajaj, C.: A structure tensor approach for 3d image skeletonization: Applications in protein secondary structure analysis. Image Processing, 2006 IEEE International Conference, pp. 2513–2516 (8–11 Oct. 2006)
30. Yushkevich, P., Piven, J., Hazlett, H., Smith, R., Ho, S., Gee, J.: User-guided 3d active contour segmentation of anatomical structures: significantly improved efficiency and reliability. NeuroImage **31**, 1116–1128 (2006)
31. Zhang, H., Maslov, K., Stoica, G., Wang, L.: Functional photoacoustic microscopy for high-resolution and noninvasive in vivo imaging. Nat. Biotechnol. **24**, 848–851 (2006)

Sasakthi S. Abeysinghe is a Ph.D. candidate at the Washington University in St. Louis, USA, and received his M.Sc. in Computer Science in 2007 from the same institution. He received his B.Sc. (Hons) in Information Systems from the Manchester Metropolitan University, UK in 2004. His current research interests lie in the areas of shape modeling using geometric skeletonization, and shape understanding of biomedical images.

Tao Ju is an Assistant Professor in the Department of Computer Science and Engineering at the Washington University in St. Louis (USA). He obtained his M.Sc. and Ph.D. degrees in Computer Science at Rice University in 2005. He conducts research in computer graphics and biomedical applications, and is particularly interested in geometric modeling and processing.