Flux Driven Fly Throughs

Sylvain Bouix[†] Kaleem Siddiqi[†] Allen Tannenbaum[§]

[†]School of Computer Science
McGill University
Montreal, Canada

[§]School of Biomedical Engineering
Georgia Institute of Technology
Atlanta, Georgia, USA

Abstract

We present a fast, robust and automatic method for computing central paths through tubular structures for application to virtual endoscopy. The key idea is to utilize a medial surface algorithm which exploits properties of the average outward flux of the gradient vector field of a Euclidean distance function the boundary of the structure of interest. The algorithm is modified to yield a collection of 3D curves, each of which is locally centered. The approach requires no user interaction, is virtually parameter free and has low computational complexity. We illustrate the approach on segmented colon and vessel data.

1. Introduction

With the availability of high resolution CT and MR images of hollow structures such as the bronchial tree, the gastrointestinal tract, blood vessels and arteries, there is an increasing interest in the field of virtual endoscopy [6]. The main motivation is to complement the often painful and costly invasive procedure of endoscopy, which involves navigating through such structures in a patient with a endoscope, in order to generate views of internal surfaces for diagnostic purposes and surgical planning. Virtual endoscopy typically involves three steps: 1) the medical images are processed in order to segment the surfaces of structures of interest, 2) flight paths are generated in order to obtain potential camera viewing positions and 3) given the flight paths, perspective views of the internal surfaces of such structures are generated using conventional graphics rendering techniques.

There has been an explosion of work in the computer vision and medical imaging literature on the development of algorithms for surface segmentation. Whereas research on segmentation remains active, a number of such algorithms in the literature, e.g., those based on deformable models [7, 10, 8, 2] work well on hollow structures because they typically have high contrast boundaries.[1] Similarly, the third step in virtual endoscopy, the rendering of views, is essentially a graphics problem for which a number of software packages such as the Visualization Tool Kit (VTK) provide standard routines that can be used. However, the second step, that of generating flight paths automatically or semi-automatically, has received relatively less attention, and is the focus of the current paper.

We now review some recent related work on finding central paths in tubular structures. Deschamps and Cohen relate the problem of finding central paths to that of finding paths of least action [3] in 3D intensity images. This leads to a form of the well-known eikonal equation where a front is propagated in the image with a speed determined by a scalar potential that depends upon location in the medium. The potential function is designed to take into account a Euclidean distance function from the boundary of a tubular structure, so that the minimal paths are centered. Beyond the requirement that the user must specify the starting and endpoints of the path, the algorithm requires little user interaction. The flow is implemented using fast marching schemes and is hence computationally efficient.

In a related though discrete approach, Bitter *et al.* build a graph from a coarse approximation of a 3D skeleton. Each edge of the graph is assigned a weight which is a combination of Euclidean distance from a user defined source node and distance from the boundary of the object. The centerline is then extracted using Dijkstra's shortest path algorithm on this graph [1].

Paik *et al.* suggest a different approach where a minimal path is found on the surface of a tubular structure and is later centered [11]. The user chooses the two end points of a desired path. The surface of the object is extracted and the closest surface voxels from the user specified points are computed. A path on the surface which minimizes the distance between these two points is then found. This path is then centered in the structure by applying an iterative thin-

[1] There are of course important exceptions, such as very narrow blood vessels viewed in MR angiography.

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

ning procedure.

An approach which is most closely related to the one presented here is that of [5]. First, a 3D skeleton of the binary colon volume is generated using a fast topological thinning algorithm. Then the skeleton is pruned from loops and spurious branches using graph search techniques. The result is then modified to yield a centered path between two user specified end points.

In this paper we present an approach that is based on a recent medial surface algorithm that has a number of proven robustness properties and which has low computational complexity [13]. Our framework offers the advantage that no user interaction is required and that all the central paths through tubular structures of complex topology are found. Most of the other methods in the literature require the path endpoints to be selected and are inherently sequential – the paths are found one at a time. We present an overview of the average outward flux based medial surface algorithm in Section 2. We then modify it to yield an algorithm which extracts a central path in a tubular structure of arbitrary topology in Section 3. We illustrate the algorithm with experimental results on colon and vessel data in Section 4.

2. Average Outward Flux Based Medial Surfaces

This section presents an overview of the skeletonization algorithm introduced in [13].

2.1. The Hamilton Jacobi Formulation

Consider Blum's grassfire flow

$$\frac{\partial S}{\partial t} = \hat{\mathbf{N}} \tag{1}$$

acting on a closed 3D surface S, such that each point on its boundary is moving with unit speed in the direction of the inward normal $\hat{\mathbf{N}}$. In physics, such equations are typically solved by looking at the evolution of the phase space of an equivalent Hamiltonian system. Let D be the Euclidean distance function to the initial surface $\mathcal{S}_0$. The magnitude of its gradient, $\|\nabla D\|$, is identical to 1 in its smooth regime. With $\mathbf{q} = (x, y, z)$, $\mathbf{p} = (D_x, D_y, D_z)$ and $\|\mathbf{p}\| = 1$, the Hamiltonian system is given by

$$\dot{\mathbf{p}} = (0, 0, 0), \qquad \dot{\mathbf{q}} = (D_x, D_y, D_z) \tag{2}$$

with an associated Hamiltonian function $H = 1 + \|\nabla D\|$. The discrimination of medial from non-medial surface points can be approached by computing the "average outward flux" of the vector field $\mathbf{q}$ at a point. This quantity is given by

$$\mathrm{Flux}(\mathbf{q}) = \frac{\int_{\delta R} < \dot{\mathbf{q}}, \hat{\mathbf{N}}_{\mathbf{o}} > dS}{\mathrm{area}\ (\delta R)} \tag{3}$$

where dS is a surface area element of the bounding surface δR of a volume R and $\hat{\mathbf{N}}_{\mathbf{o}}$ is the outward normal at each point on the surface. It can be shown that as the volume shrinks to a point not on the medial surface, the average outward flux approaches zero. In contrast, when the volume over which it is computed shrinks to a medial surface point, the average outward flux approaches a strictly negative number [13]. Thus, it is an effective way for distinguishing between these two cases. This calculation is discretized, as described in Algorithm 1, and is used to guide a thinning process in a cubic lattice, while taking care to preserve the object's topology.

Algorithm 1: Average Outward Flux.

Compute the distance transform D of the object ;
Compute the gradient vector field ∇D;
Compute the average outward flux of ∇D using Eq. 3;
for *(each point* $\mathbf{x}$ *in the interior of the object)* **do**

$$\mathrm{Flux}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{26} < \hat{\mathbf{N}}_{\mathbf{i}}, \nabla D(\mathbf{x}_i) >;$$

(where $\mathbf{x}_i$ is a 26-neighbor of $\mathbf{x}$ and $\hat{\mathbf{N}}_{\mathbf{i}}$ is the outward normal at $\mathbf{x}_i$ of the unit sphere centered at $\mathbf{x}$)

2.2. Preserving Topology

A point is a *simple* point if its removal does not change the topology of the object. Hence in 3D, its removal must not disconnect the object, create a hole, or create a cavity. Malandain *et al.* have introduced a topological classification of a point $\mathbf{x}$ in a cubic lattice by computing two numbers, [9]:

- C^*: the number of 26-connected components 26-adjacent to $\mathbf{x}$ in $\mathrm{O} \cap \mathrm{N}^*_{26}$
- $\bar{C}$: the number of 6-connected components 6-adjacent to $\mathbf{x}$ in $\bar{\mathrm{O}} \cap \mathrm{N}_{18}$.

Where O is the 26-connected object, N^*_{26} is the 26 neighborhood of $\mathbf{x}$ without $\mathbf{x}$ and N_{18} is the 18 neighborhood of $\mathbf{x}$ including $\mathbf{x}$. Further, they have shown that if $C^* = 1$ and $\bar{C} = 1$, the point is *simple*, and hence removable.

The basic strategy now is to guide the thinning of the object by the average outward flux measure computed over a

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

very small neighborhood. Points with the most negative average outward flux are the strongest medial surface points. The process is stopped when all surviving points are not simple, or have an average outward flux below some chosen (negative) value, or both. Unfortunately the result is not guaranteed to be a thin set, i.e., one without an interior.

This last constraint can be satisfied by defining an appropriate notion of an endpoint in a cubic lattice. In $\mathbb{R}^3$, if there exists a plane that passes through a point $\mathbf{x}$ such that the intersection of the plane with the object includes an open curve which ends at $\mathbf{x}$, then $\mathbf{x}$ is an end point of a 3D curve, or is on the rim or corner of a 3D surface. This criterion can be discretized easily to 26-connected digital objects by examining 9 digital planes in the 26-neighborhood of $\mathbf{x}$, [12]. The thinning process proceeds as before, but the threshold criterion for removal is applied only to endpoints, as described in Algorithm 2.

Algorithm 2: Topology Preserving Thinning: Medial Surface.

for *(each point* $\mathbf{x}$ *on the boundary of the object)* **do**
  if *(*$\mathbf{x}$ *is simple)* **then**
    insert($\mathbf{x}$, maxHeap) with $\mathrm{Flux}(\mathbf{x})$ as the sorting key for insertion;
while *(maxHeap.size* > 0*)* **do**
  $\mathbf{x}$ = HeapExtractMax(maxHeap);
  if *(*$\mathbf{x}$ *is simple)* **then**
    if *(*$\mathbf{x}$ *is an end point) and (*$\mathrm{Flux}(\mathbf{x}) < Thresh$*)* **then**
      mark $\mathbf{x}$ as a medial surface (end) point;
    else
      Remove $\mathbf{x}$;
      for *(all neighbors* $\mathbf{y}$ *of* $\mathbf{x}$*)* **do**
        if *(*$\mathbf{y}$ *is simple)* **then**
          insert($\mathbf{y}$, maxHeap) with $\mathrm{Flux}(\mathbf{y})$ as the sorting key for insertion;

2.3. Labeling the Medial Surface

The medial surface can be labeled using the classification of Malandain *et al.* [9]. Specifically, the numbers C^* and $\bar{C}$, described in Section 2.2, can be used to classify curve points, surface points, border points and junction points. The complete topological classification is presented in Table 1. However, junction points can be misclassified as surface points when certain special configurations of voxels occur, and these cases have to be dealt with using a new definition for simple surfaces [9].

$\bar{C}$	C^*	Type
0	any	interior point
any	0	isolated point
1	1	border (simple) point
1	2	curve point
1	> 2	curves junction
2	1	surface point
2	> 2	surface-curve(s) junction
> 2	1	surfaces junction
> 2	≥ 2	surfaces-curves junction

Table 1. The topological classification of Malandain et al.

Let $\mathbf{x}$ be a surface point ($\bar{C} = 2$ and $C^* = 1$). Let $\mathrm{A}_{\mathbf{x}}$ and $\mathrm{B}_{\mathbf{x}}$ be the two connected components of $\bar{\mathrm{O}} \cap \mathrm{N}_{18}$ 6-adjacent to $\mathbf{x}$. Two surface points $\mathbf{x}$ and $\mathbf{y}$ are in an equivalence relation if there exists a 26-path $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_n$ with $\mathbf{x}_0 = \mathbf{x}$ and $\mathbf{x}_n = \mathbf{y}$ such that for $i \in [0, ..., n-1]$, ($\mathrm{A}_{\mathbf{x}_i} \cap \mathrm{A}_{\mathbf{x}_{i+1}} \neq \emptyset$ and $\mathrm{B}_{\mathbf{x}_i} \cap \mathrm{B}_{\mathbf{x}_{i+1}} \neq \emptyset$) or ($\mathrm{A}_{\mathbf{x}_i} \cap \mathrm{B}_{\mathbf{x}_{i+1}} \neq \emptyset$ and $\mathrm{B}_{\mathbf{x}_i} \cap \mathrm{C}_{\mathbf{x}_{i+1}} \neq \emptyset$). A *simple surface* is then defined as any equivalence class of this equivalence relation. We use this definition in our framework to find all the misclassified junctions. If the 26-neighborhood of a previously classified surface point $\mathbf{x}$ is not a *simple surface*, then $\mathbf{x}$ is a junction point.

3. Extracting Central Paths

We now develop a framework to compute central paths in single tubular 3D structures (such as colons) as well as tree-like objects that are composed of branches that are tubular (such as vessels). The underlying motivation is that the medial surface of a cylindrical structure approaches a 3D curve centered in the object's interior. Our strategy is to thin the medial surface to obtain a structure composed of only curves (the medial curve) and to then prune the result to obtain well centered paths. Assuming that we have an object with one connected component we now describe the process to obtain the medial curve from a medial surface.[2]

3.1. The Medial Curve

The medial surface extraction algorithm (algorithm 2) can be easily modified to obtain a medial curve. In fact, only the end point criterion needs to be changed. Recall that during the thinning, if a point $\mathbf{x}$ is an end point of a 3D curve, or on the rim or corner of a 3D surface, it will not

[2] If one had more than one connected component the same process could be applied to each component in turn.

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

be removed if its average outward flux is strongly negative. Now, if only end points of curves are preserved, every surface point will be removed and the thinning will lead to a medial curve centered in the object. In a cubic lattice, an end point of a 26-connected curve is defined as a point $\mathbf{x}$ whose 26-neighborhood $\mathrm{O} \cap \mathrm{N}_{26}^{*}$ only contains one object point.

It is also preferable to order the thinning by distance and not by average outward flux in order to obtain well centered paths, because several neighboring medial points on the same simple surface can have the similar flux values. Algorithm 3 presents the thinning procedure to extract the medial curve. Note that only two steps have been modified to algorithm 2: the sorting key during insertion of new points in the heap and the end point criteria. Figures 1 and 2 show the medial surface and medial curve of a colon data set and a data set of coronary arteries, respectively.

Algorithm 3: Topology Preserving Thinning: Medial Curve.

for *(each point* $\mathbf{x}$ *on the boundary of the object)* **do**
  if *(*$\mathbf{x}$ *is simple)* **then**
    insert($\mathbf{x}$, maxHeap) with $-D(\mathbf{x})$ as the sorting key for insertion;
while *(maxHeap.size* > 0*)* **do**
  $\mathbf{x}$ = HeapExtractMax(maxHeap);
  if *(*$\mathbf{x}$ *is simple)* **then**
    if *(*$\mathbf{x}$ *is an end point of a 3D curve) and (*$\mathrm{Flux}(\mathbf{x}) <$ *Thresh)* **then**
      mark $\mathbf{x}$ as a medial curve (end) point;
    else
      Remove $\mathbf{x}$;
      for *(all neighbors* $\mathbf{y}$ *of* $\mathbf{x}$*)* **do**
        if *(*$\mathbf{y}$ *is simple)* **then**
          insert($\mathbf{y}$, maxHeap) with $-D(\mathbf{y})$ as the sorting key for insertion;

3.2. Pruning

Once the medial curve is extracted some pruning needs to be done. The pruning method depends on the type of output desired. If the goal is to extract one centerline path, as is the case for the colon, one can use Dijkstra's single source shortest path algorithm [4]. The idea is to compute the shortest path from every endpoint on the medial curve to every other endpoint and pick the longest shortest path as the centerline path. This procedure is described in algorithm 4.

Algorithm 4: A single centerline path.

Label the medial curve into endpoints, curve points and branch points;
max = $-\infty$;
for *(all endpoints)* **do**
  Compute the single source shortest path of the medial curve from the endpoint;
  Find Longest shortest path c;
  if *(length(*c*)* $>$ *max)* **then**
    max = length(c);
    centerline = c;

If one wants to extract a centerline tree, as is the case for blood vessels or airways, one can use a traditional pruning: remove all branches connected to endpoints whose length is lower than some threshold and repeat the operation until every branch is long enough. The details are given in algorithm 5. In the case of arteries and airways, the pruning is very stable as the diameter of a vessel is typically much smaller than its length. Hence, the results are not sensitive to the choice of this length parameter.

Algorithm 5: A centerline tree.

repeat
  Label the medial curve into endpoints, curve points and branch points;
  Compute length of branches connected to endpoints;
  for *all branches* **do**
    if *length(branch)* $<$ *Thresh* **then**
      remove branch from medial curve;
until *all branches have length* $>$ *Thresh*;

Note that it is also possible to extract a centerline tree using the shortest path method, by selecting the longest paths one by one until all the remaining paths have lengths below some threshold or until a predetermined number of longest paths has been extracted. Similarly, one can prune the medial curve until only one path is left.

4. Experiments

4.1. Data Set

We have tested our central path algorithms on a $512 \times 512 \times 400$ CT colon data set provided by the Surgical Planning Laboratory of Brigham and Women's Hospital, on a

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

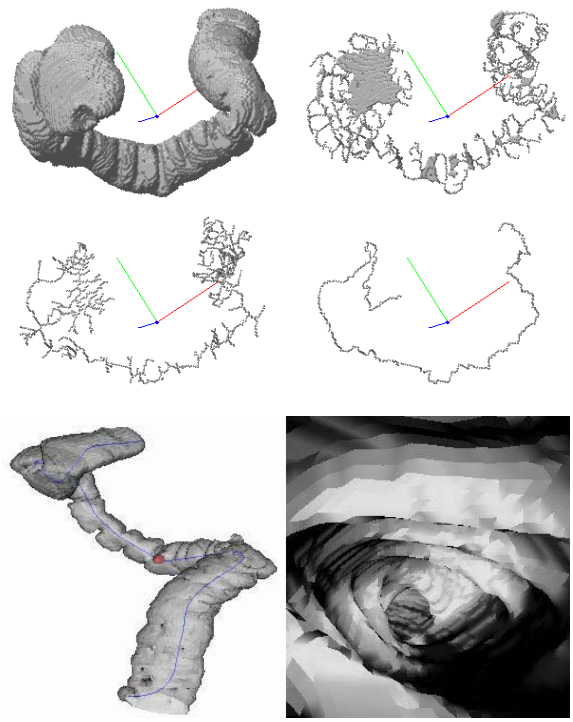

Figure 1. From Left to Right, Top to Bottom: Segmented Colon, Medial Surface, Medial Curve, Centerline Path, Smooth Path and Inside View.

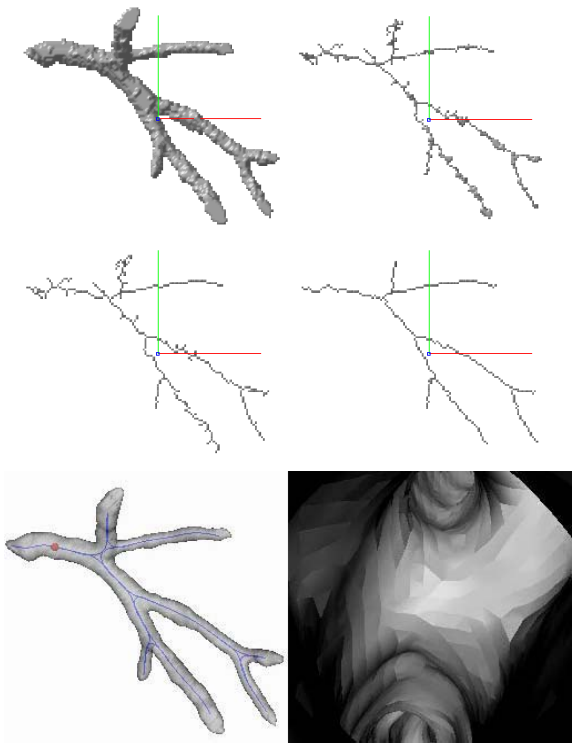

Figure 2. From Left to Right, Top to Bottom: Segmented Arteries, Medial Surface, Medial Curve, Centerline Tree, Smooth Path and Inside View.

$515 \times 512 \times 286$ CT data set of coronary arteries provided by the Cleveland Clinic and on a $360 \times 330 \times 420$ computed rotational angiography (CRA) data set of the head from the John P. Robarts Research Institute.

4.2. Surface Segmentation

The CRA data set was segmented using the flux maximizing flow algorithm [14] and conformal snake algorithms based on [2, 8] were used in the segmentation of the colon and coronary arteries. These latter two data sets were segmented by employing discretizations motivated partially by the work of Brakke, which are discussed in detail in [15].

4.3. Results

A single centerline path through the colon data set was obtained by applying algorithms 1, 3 and 4. Figure 1 shows the original object, the medial surface (alg. 2), the medial curve (alg. 3), the centerline path (alg. 4) and a screen shot of a fly through movie **colon.mpg**, which is available at **http://www.cim.mcgill.ca/˜sbouix/research/data/movies**. A centerline tree for the arteries was obtained using algorithms 1, 3 and 5. The fly through path was extracted by performing a Depth First Search of the tree. Note that this method would work with objects of arbitrary topology. Figure 2 shows the arteries, the medial surface (alg. 2), the medial curve (alg. 3), the centerline tree (alg. 5) and a screen shot of a second fly through movie **arteries.mpg**, which is also available at **http://www.cim.mcgill.ca/˜sbouix/research/data/movies**. The results of the procedure applied to cropped images from segmented CRA data are displayed in figure 3. Observe that the complex topology (cycles and branches) and geometry of the data are captured accurately by the extracted paths.

In all cases, the flux threshold was chosen so that 25% of the voxels had higher magnitude average outward flux values and the branch length threshold was determined by the expected radius of the narrowest tubular structure. The computed centerlines were smoothed using a cubic spline and the fly through movies were computed using VTK rou-

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE

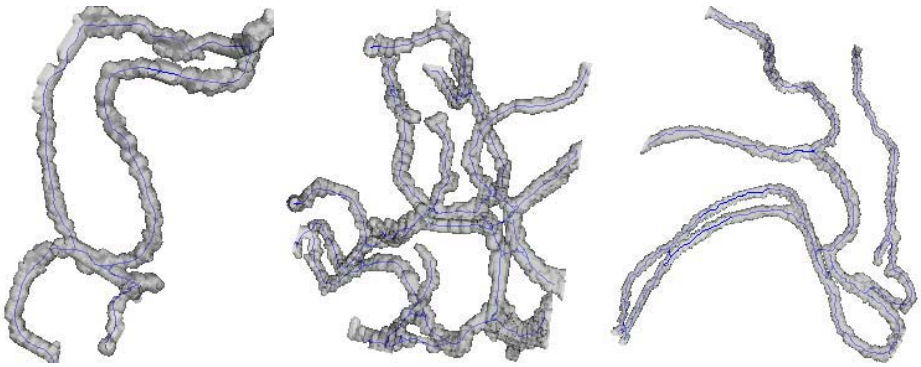

Figure 3. Central paths in cropped regions of a segmented (CRA) data set of the head. Note the complexity of the structures and the accuracy of the paths.

tines. The position and orientation of the camera was either set to follow the frenet frame of the 3D curve or to try to keep the view up vector as consistent as possible.

5. Conclusion

We have presented a robust and automatic method for finding central paths through tubular structures. The method enjoys a number of the advantages of the medial surface algorithm on which it is based including accuracy, robustness against noise and low computational complexity ($\mathcal{O}(k \log k)$, where k is the number of voxels in the object) [13]. Furthermore, it requires no user interaction and is essentially parameter free as the flux threshold in the computation of the medial curve and the branch length threshold in the pruning of the centerline tree are computed automatically. We have illustrated the method with a single centerline computed for a colon data set, a centerline tree computed for a data set of coronary arteries and the associated fly through movies for each. We have also provided examples of central paths in complex CRA vessel data.

References

[1] I. Bitter, A. E. Kaufman, and M. Sato. Penalized-Distance Columetric Skeleton Algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):195–206, 2001.

[2] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic snakes. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[3] T. Deschamps and L. D. Cohen. Fast Extraction of Minimal Paths in 3D Images and Applications to Virtual Endoscopy. *Medical Image Analysis*, 5(4):281–299, December 2001.

[4] E. W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1:269–271, 1959.

[5] Y. Ge, D. R. Stelts, J. Wang, and D. Vining. Computing the Centerline of a Colon: a Robust and Efficient Method Based on 3D Skeletons. *Journal of Computer Assisted Tomography*, 23(5):786–794, 1999.

[6] F. A. Jolesz, W. E. Lorensen, H. Shinmoto, H. Atsumi, S. Nakajima, P. Kavanaugh, P. Saiviroonporn, S. E.Seltzer, S. G. Silverman, M. Phillips, and R. Kikinis. Interactive Virtual Endoscopy. *American Journal of Radiology*, 169:1229–1237, 1997.

[7] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1987.

[8] S. Kichenassamy, P. Olver, A. Tannenbaum, and A. Yezzi. Conformal curvature flows: from phase transitions to active vision. *Archive of Rational Mechanics and Analysis*, 134:275–301, 1996.

[9] G. Malandain, G. Bertrand, and N. Ayache. Topological Segmentation of Discrete Surfaces. *International Journal of Computer Vision*, 10(2):183–197, 1993.

[10] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, February 1995.

[11] D. S. Paik, C. F. Beaulieu, G. D. R. Brooke Jeffrey, Rubin, and S. Napel. Automated Path Planning for Virtual Endoscopy. *Medical Physics*, 25(5):629–637, May 1998.

[12] C. Pudney. Distance-Ordered Homotopic Thinning: A Skeletonization Algorithm for 3D Digital Images. *Computer Vision and Image Understanding*, 72(3):404–413, 1998.

[13] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker. Hamilton-Jacobi Skeletons. *International Journal of Computer Vision*, 48(3):215–231, 2002.

[14] A. Vasilevskiy and K. Siddiqi. Flux Maximizing Geometric Flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12), December 2002.

[15] A. Yezzi and A. Tannenbaum. 4d active surfaces for cardiac analysis. In *MICCAI'02*, 2002.

Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)
1063-6919/03 $17.00 © 2003 IEEE