Extracting Skeletal Curves from 3D Scattered Data

Anne Verroust
INRIA Rocquencourt
Domaine de Voluceau, B.P. 105
78153 Le Chesnay Cedex, FRANCE
Anne.Verroust@inria.fr

Francis Lazarus
I.R.C.O.M.-S.I.C. SP2MI
Boulevard 3, Teleport 2, B.P. 179
86960 Futuroscope Cedex, FRANCE
lazarus@sic.univ-poitiers.fr

Abstract

We introduce a method for extracting skeletal curves from an unorganized collection of scattered data points lying on a surface. These curves may have a tree like structure to capture branching shapes such as blood vessels. The skeletal curves can be used for different applications ranging from surface reconstruction to object recognition.

1 Introduction.

The description of objects with cylindrical shapes has been extensively used in the computer vision community [1, 32, 30, 35]. In a first approximation it is often convenient to model a shape as a generalized cylinder. This is particularly relevant in the context of reconstruction of anatomical shapes from CT data or scanner [27, 34, 23]. Numerous works have been done for object description using tree-like structures [29, 24, 25] or cylindrical representations [11]). What makes the use of cylindrical descriptions and generalized cylinders so popular is their relative simplicity. A generalized cylinder is parameterized by a spin curve together with a set of cross sections [1]. Depending on the application, the recovery of such parameters usually requires less work than it would be for a more general description of a shape.

Note that a spin curve can be used as a skeleton of an implicit surface model [8, 7]: the surface of the object is defined as an iso-surface of a field function generated by the skeleton. In [6, 14] the authors propose an implicit reconstruction of shapes from a set of 3D scattered points using the Voronoï graph of the data points to built a geometric skeleton. Our method is an alternate skeleton construction. It avoids the computation of the 3D Voronoï graph.

Other applications start with volume images, so that the traditional medial axis transform can be applied [23].

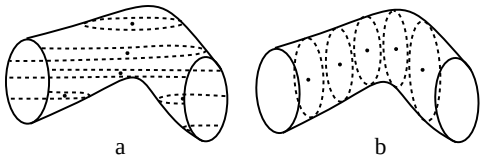

Figure 1. Two sets of 2D sections sampled on a surface.

As opposed to voxelized data, we start with an unorganized cloud of points sampled on a surface. A 3D curve must be specified inside the cloud of points in order to use a cylindrical model and to represent the surface. When the set of 3D points is given as a set of 2D planar scans, Burdin et al. [11] use the curve defined by the centroids of each 2D section. However, this implies a consistent orientation of the cross-sectional planes relatively to the position of the object. Figure 1 shows two different orientations of the cross-sectional planes. Clearly, configuration (a) is not adapted. For some objects, because of their bend shape , it is even impossible to find a suitable orientation for the sectional planes.

Although essential, the issue of finding a suitable axis in order to design a generalized cylinder was rarely discussed. In practical cases, this task can be tedious as mentioned by Nazarian & al. [24]. These authors propose two methods to automate the determination of the axis:

- the first one consists in a recursive subdivision of the set of data points. At each step the subsets of data points are split by a plane perpendicular to their main axis of inertia and passing through their barycenter. The resulting axis is a polyline composed of main axes of parts of the initial set of data points (cf. Figure 2.a). The authors recognize their method does not always work and show a counterexample where the main axis of inertia, represented in dashed line in Figure 2.b, cannot be used to find the central axis.
- the second method works with a set of data points given

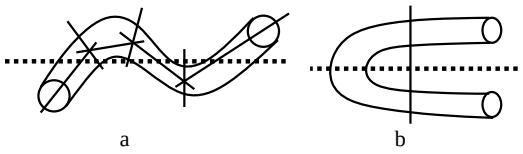

Figure 2. a: recursive subdivision using main axes of inertia (Nazarian & al); the first main axis is shown in dashed line; The main axes of inertia and the perpendicular splitting planes for the second level of recursion are shown in plain lines. b: The first axis of inertia and the perpendicular splitting plane are not suitable for the subdivision algorithm.

by planar 2D scans. It uses the triangulation computed by Boissonnat's method [10] as an approximation of the tubular shape.

In this paper, we present a method that automatically computes a set of 3D curves in a cluster of points. When the points lie on a surface these curves may be used as axes for a cylindrical approximation of the surface. These curves may form a tree structure if the surface resemble a "tree of generalized cylinders". The method works independently of the bending of the surface. As a consequence it will return a correct result for the shape shown on Figure 2.b. Like in other reconstruction algorithms [4, 21, 2], we make some hypothesis on the input cluster of points.

1. The data points should be sampled on a certain (unknown) surface.

2. The sampling should be done in such a way that the distance between neighbor points is small with respect to the width of the tubular parts of the surface.

The computation of the axes is based on a decomposition of the surface to be reconstructed into level sets of a scalar function. In [18, 19, 20] we used a similar decomposition applied to polyhedral surface. In this work a simplified discrete version of the Morse theory was developed (see [31, 17, 33, 16] for use of Morse Theory in Graphics).

In the present case, we are dealing with a cluster of points and no topological information, such as adjacency relationship, is available. We cannot apply the method of [18, 19, 20] which uses this information to build the decomposition of the surface.

Here, our goal is to compute the axes rapidly, rather than reconstructing the surface. We follow the main steps of [18, 19] and adapt them to our particular case. The scalar function used in the decomposition is an approximation of a geodesic distance function to a given point. This function is determined by a single point from the set of data points called the *source point*. The source point may be interactively designated by the user. It may also be automatically selected using a simple heuristic as explained at the end of Section 3. Intuitively, the source point corresponds to a polar extremity of the shape as shown on Figure 3. Once the

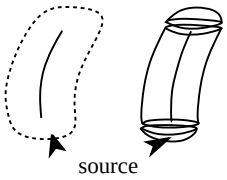

Figure 3. The source point and the corresponding decomposition.

decomposition into level sets of a scalar function is done, a single axis is obtained for each cylindrical part associated to it. .

2 Overview

The computation of the 3D curves is made in four steps, as illustrated in Figure 4. Given a set of data points,

1. We build the *neighborhood graph* of order n. This is a non oriented graph whose vertices are the input data points. The edges have the form (v,v'), where the vertex v' or the vertex v is one of the n nearest neighbors of respectively v or v'. The subgraphs of the neighborhood graph induced by a point and its neighbors may be non planar and, in general, it will not be possible to fit a suitable polygonal surface to this graph.

2. From this neighborhood graph and a source point, we construct a distance map and an oriented subgraph called the *geodesic graph*. The geodesic graph is obtained by computing shortest paths of the neighborhood graph between all vertices and the source point. The distance map is simply the length of these shortest paths.

The construction of the neighborhood graph, the distance map and the geodesic graph are described in details in Section 3.

3. We further compute k level sets of the distance map using the neighborhood graph, where k is a user defined value. The level sets are composed of points at a constant distance from the source. We connect these points on a proximity criterion and identify the connected components. We also compute the centroids of each of these connected components.

4. The axes are finally obtained by connecting the centroids of successive levels as previously computed. When the levels contain several connected components, the connection of the centroids are based upon the paths of the geodesic graph.

Steps 3 and 4 are discussed in Section 4 and examples are provided in Section 5.

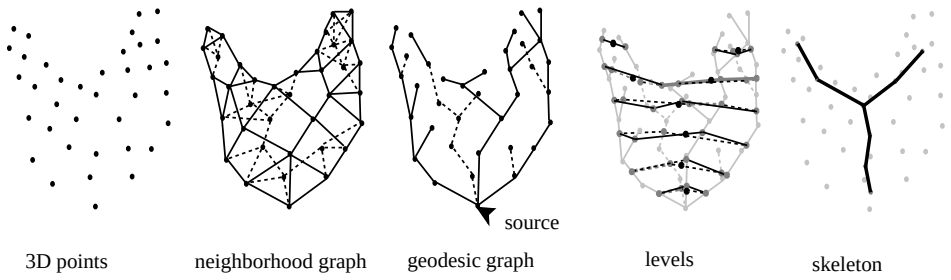

Figure 4. The main steps of our method.

3 The Geodesic Graph

We are given a set of data points $X = \{x_1, ..., x_m\}$ sampled on an unknown surface $\mathcal{S}$ and an integer n. The neighborhood graph $\mathcal{N}_{(X,n)}$ is an undirected graph defined as follow:

1. the vertices of $\mathcal{N}_{(X,n)}$ are the points of X

2. a pair (x_i, x_j) is an edge of $\mathcal{N}_{(X,n)}$ if either x_i is one of the n nearest points of x_j in X or x_j is one of the n nearest points of x_i in X.

In order to compute the n nearest points of each vertex, we use a subdivision of the space into voxels. We subdivide the enclosing box of the data points into a regular grid of voxels and associate each data point with the voxel containing it. The search of the n nearest neighbors of a data point x is made in two steps. We first visit the voxels close to the one containing x progressively moving away until n points (v_1,...v_n) are found. Then, we put $d = \max_{1 \leq i \leq n}(\text{distance } (v_i, x))$ the maximum distance between x and these n points. As we have already found n points (v_1,...v_n) located at a distance lower or equal to d from x, we are sure that the n nearest neighbors of x belong to the sphere $S(x, d)$ with center x and radius d. We search all the data points belonging to $S(x, d)$ and we extract the n nearest points on this set, using a quick sort on the distance values.

The number n should be chosen so that $\mathcal{N}_{(X,n)}$ is a connected graph. This choice depends on the regularity of the sampling step of the data points on the surface. In the example presented in Figure 8 we used $n = 5$. When the data points are not uniformly spread over the surface, larger values may be necessary. For instance, in the case of 2D planar scans, where the resolution is smaller inside the scan planes than between the cross-sections, we found that a value of $n = 10$ was necessary to obtain a connected graph. This was the case for the blood vessels shown in Figure 10.

Together with the selected source point x_s, the neighborhood graph is used to compute the geodesic graph $\mathcal{G}_{(X,n,x_s)}$. The geodesic graph is composed of geodesic paths of the neighborhood graph emanating from the data points and joining the source point. The length of an edge of the neighborhood graph is taken as the Euclidean distance between its endpoints. We use Dijkstra's algorithm [12] to efficiently compute these paths and their length. Precisely, Dijkstra's algorithm compute the geodesic distance from any point to the source as well as a pointer to the next point in a geodesic path joining the source. Note that there may exist several geodesic paths from a point to the source. We simply use the path returned by Dijkstra's algorithm. The distance map introduced in Section 2 is defined for every data point to be its geodesic distance to the source. It is extended to the interior of the edges of the geodesic graph with a linear interpolation of the endpoint distance values of the edges.

As we supposed that the edge of the neighborhood graph should lie on the surface, the geodesic paths previously computed are intended to approximate geodesic paths on the surface. Note that The computation of geodesic paths for polyhedral surfaces have been the subject of numerous publications [22, 26]. However, we cannot apply those methods since our data points do not have a polyhedral structure.

Note that $\mathcal{G}_{(X,n,x_s)}$ is a connected subgraph of $\mathcal{N}_{(X,n)}$ since all the geodesic paths share the source point. $\mathcal{G}_{(X,n,x_s)}$ is actually a tree since Dijkstra's algorithm selects only one path from any vertex to the source. An example of a geodesic graph is shown Figure 6 for a set of data points registered on a femur[1].

The source point x_s may be selected by the user or automatically computed by the following two step heuristic:
$-1-$ a point x_0 is chosen at random (e.g. the first point in the data set).
$-2-$ x_s is defined as the furthest point from x_0 on the neighborhood graph.

When the object has a "tubular structure", even with branches, the furthest point from x_0 is a point located at the extremity of a tubular part of the surface. This point can be obtained as the last visited vertex when running Dijkstra's algorithm a first time with source point x_0.

[1] data, courtesy of Jai Menon, IBM T. J. Watson Research Center.

4 Levels and branches

After computing the geodesic graph and the distance map, we extract k level sets. Intuitively, each level set corresponds to a set of "sections" of the decomposition of the surface given by the set of data points $X = \{x_1, ..., x_m\}$. The number k is given by the user to specify the number of points used to approximate the longest axis. In practice k can be chosen as the ratio of the distance of the furthest point to the average edge length. The k levels d_1,..., d_k are uniformly distributed over the interval $[\alpha d_{max}, \beta d_{max}]$, where d_{max} is the extremum of the distance map and α and β are two percentage coefficients. We used $\alpha = 3\%$ and $\beta = 97\%$ on Figure 11. The ith level set $\mathcal{S}_i$ is the set of points on the neighborhood graph whose (extended to edge interiors) distance is d_i.

Next, we need to partition each level set $\mathcal{S}_i$ into subsets corresponding to the different branches of the surface.

Here, we cannot proceed as in [18, 19] where each vertex of the polyhedron can be classified as regular or critical according to the configuration of distances of its neighbors (see Figure 5). In the polyhedral case, the decomposition is

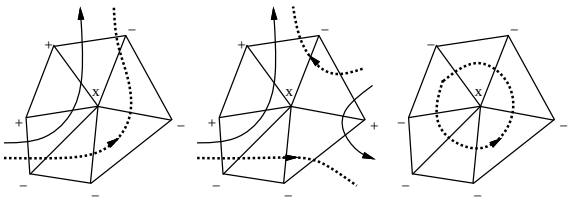

Figure 5. the leftmost vertex is regular and the other are critical (the $+$ (respectively $-$) associated to a vertex indicates that the geodesic distance between the vertex and the source is greater (respectively lower) than the geodesic distance between the source and x).

obtained by cutting through the level sets associated to the critical points. Also, the level sets of the geodesic distance are efficiently extracted using the adjacency between faces of the polyhedron.

In our case this adjacency relationship is not available. As a result, points cannot be classified as regular or critical which would in turn provide the branching points and the partition of each level set corresponding to the different branches. To obtain this partition, we first construct the neighborhood graph $\mathcal{N}_{(\mathcal{S}_i,2)}$, linking every point of $\mathcal{S}_i$ with its two closest neighbors. Then, we remove from $\mathcal{N}_{(\mathcal{S}_i,2)}$ the edges longer than a certain length w_i. w_i is determined using the hypothesis (2) on the data points: the distance between points is small with respect to the width of the tubular parts. In practice w_i is computed using the median of the distribution of the distance values of the second nearest neighbor of each point inside $\mathcal{S}_i$. Finally, we partition this graph into its connected components. We also compute the centroid of each connected component by taking the barycenter of its points.

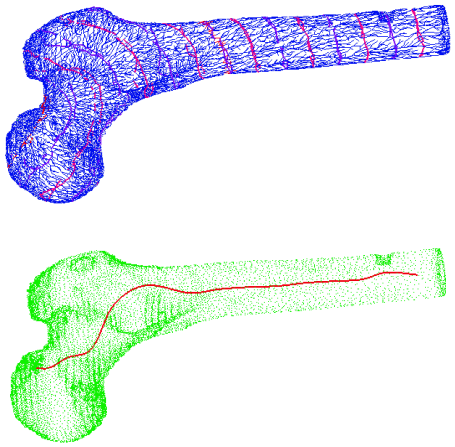

Figure 6. Up: 15 level sets computed on the geodesic graph of a femur. Down: the corresponding axis.

The skeletal curves are constructed by connecting the centroids of successive level sets. When the level sets have several connected components we use the geodesic paths to connect the centroids: if a level set $\mathcal{S}_i$ is partitioned into l_i connected components $(\mathcal{S}_{i,1}, ..\mathcal{S}_{i,l_i})$, for each $j \leq l_i$, we select a point of $\mathcal{S}_{i,j}$ and follow its geodesic path to the source point until an edge containing a point of $\mathcal{S}_{i-1}$ is encountered. This point belongs to the connected component of $\mathcal{S}_{i-1}$ which is either associated to the branch containing $\mathcal{S}_{i,j}$ or located at the birth of this branch.

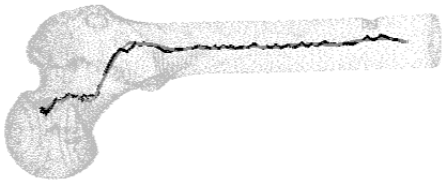

Figure 7. Different number of levels used to compute the axis of the femur : 30 (grey curve) and 90 (black curve).

Eventually, we may approximate the axes with spline curves as shown for the femur on Figure 6 and 7 . We use a simple fitting procedure from [28, 13] to compute these splines from the connected centroids.

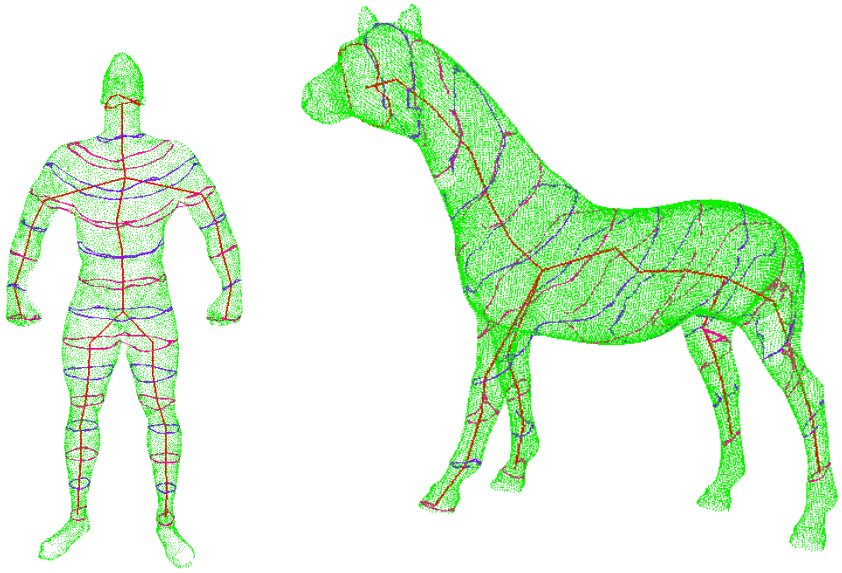

Figure 8. 15 level sets and the corresponding skeletal curves on a man and on a horse.

5 Results

We applied our skeletonization method on a set of examples presented from Figure 8 to Figure 11:

• Figure 8 shows two models of a man and a horse. These models are public domain models found on the Web pages of Cyberware[2].
The source point of the man is located at the top of his head. The horse was scanned from many orientations resulting in 210 linear scans which consisted of a series of scans of different regions (views) of the object. The source point is located at the extremity of its jaw.

Here, the branches are correctly located and a tree-like axial structure is built.

• The aorta presented in Figure 9 is described by a set of 2D scans. This data comes from the National Library of Medicine Visible Human Male [3]. Our method is little sensitive to the location of the source points: in Figure 9.A, two neighbor points and in Figure 9.B, three points located at different extremities of the tubular parts of the surface have been used to compute the axial structures. The difference between their corresponding axial structures is negligible.

• We obtained good results with our method for the two blood vessels of Figure 10[4]. These data sets are especially challenging since they contain many vessels close to each other.

• The axis of the colon presented in Figure 11 cannot be found using Nazarian's subdivision method [24]. It is described by a set of 2D scans and comes from the National Library of Medicine Visible Human Male.
Table 1 shows the CPU time required to compute the axial structure for the models shown Figures 6 to 11. We run non optimized code on a Indy The computing time depends mainly on the number of data points and on the order of neighborhood graph. In fact, the computation of the neighborhood graph is one of the most expensive part of our method.

[2] http://www.cyberware.com/models/index.html
[3] http://www.mayo.edu/bir/Models/Visible_Human_Male/VRML/index.html
[4] Data, courtesy of Jacques Feldmar.

6 Conclusion

We have presented a new technique to associate an axial structure to a set of scattered data points.
The interaction is reduced to the selection of a source point and can be automated using a simple heuristic. The main steps for the construction of the axes are:

– 1 – the computation of the neighborhood and geodesic graphs, using Dijkstra's algorithm,

– 2 – the computation of geodesic levels and the location of branching nodes.

Our method is rapid and robust with respect to the location of the source point. The "quality" of the resulting skeletal curves mainly depends on the appropriate choice of the order of the neighborhood graph.

Our skeletonization algorithm relies on the location of the source point. This means that the axial structure found by the algorithm is not unique for a given object. The axial structure is actually defined for a pair (object, source point).

The question that arises from this observation is how can we extract interesting axis for objects with tubular shape? We should note that the precise definition of a tubular shape

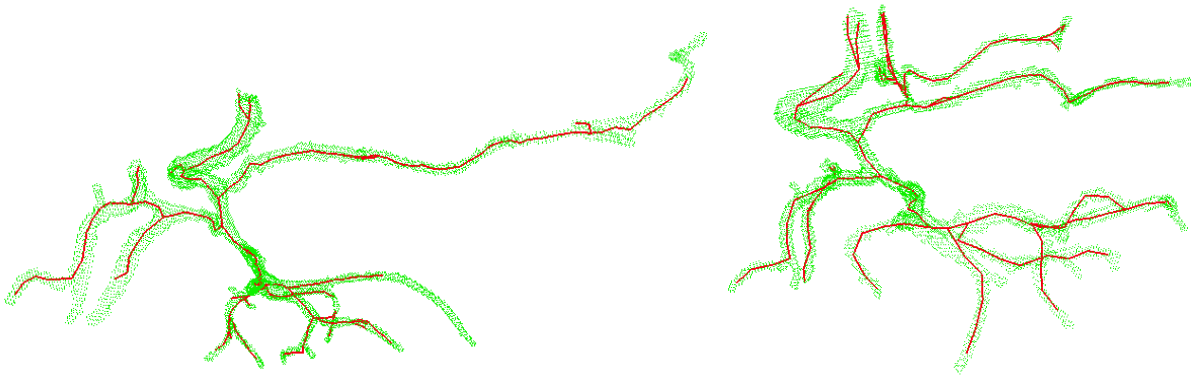

Figure 10. Two ramifications of blood vessels.

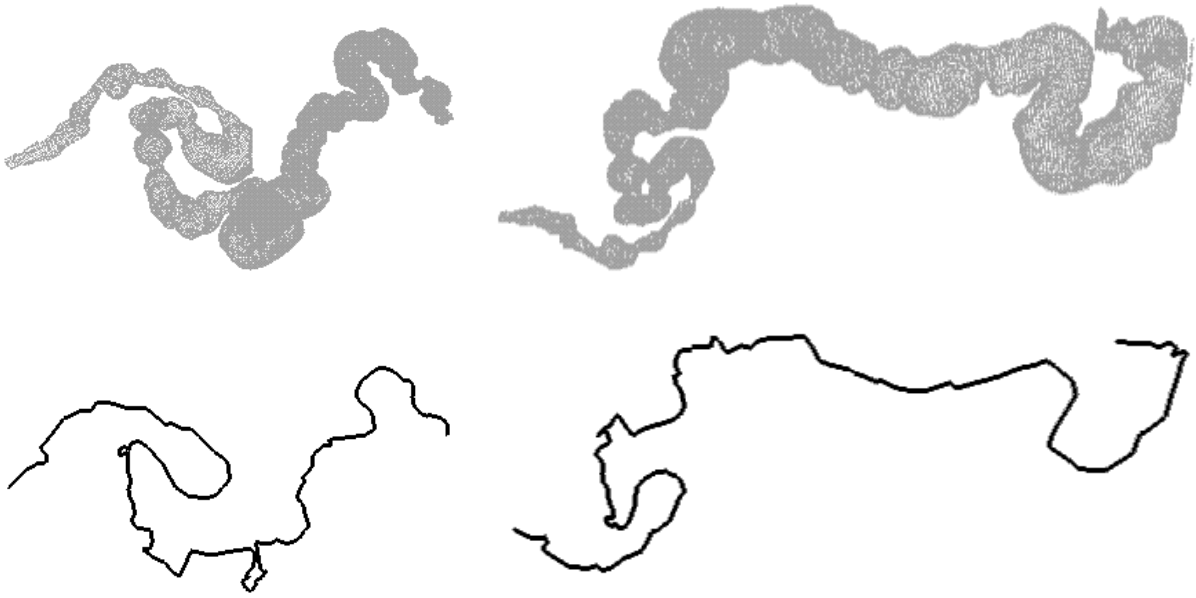

Figure 11. Up: a colon. Down: its axis (two different views).

reference of figures	number of data points	number of levels	order of the neighborhood graph	CPU time in seconds on an Indy
6	16781	15	5	27.83
7	16781	30	5	33.20
7	16781	90	5	44.83
8	21500	15	5	52.21
8 (horse)	48486	15	6	132.61
9 (A and B)	7688	50	5	40.01
9 (A)	7688	50	5	39.54
9 (B middle)	7688	50	5	35.09
9 (B right)	7688	50	5	35.91
10 (left)	11807	60	10	28.85
10 (right)	13582	50	8	45.49
11	55573	150	5	235.56

Table 1. Statistics for Figures 6 to 11.

is not clear (except for the trivial case of a curve extrusion along an axis). As a consequence there is no precise definition for the axis of a tubular shape. The medial axis, for instance, does not always produce what we would expect from a tubular object. As opposed to our skeletons the medial axis is in general composed of pieces of curves as well as pieces of surfaces with many ramifications [9, 15] It can be simplified to obtain a wireframe skeleton [3, 5] but this simplification process is not entirely automatic, as some thresholds must be fixed. The whole process may also be expensive as the simplification process works on the medial axis of the object represented by the set of sampled data points.

There are many objects, however, for which the definition of an axial structure is intuitively so obvious that we could draw this axis without any hesitation. Blood vessels are good examples of such objects. Our initial question is now turned into the more pragmatic question: how does this skeletonization algorithm provide an efficient way for computing axis in blood vessels structure? The only point we have to worry about is the selection of the source point since the computation is entirely automatic once a source point is selected. On one hand it is seen in Figure 9.A that the algorithm is little sensitive to the precise location of the source point. This is due to the continuity of the distance function with respect to the source location. On the other hand we have seen Section 3 that the source point can be selected using a simple heuristic. Figure 9.B shows that this heuristic works well in practice: extremities on different branches give rise to nearly identical axis.

Work in progress include attempts to model more precisely the location of the branching nodes. A more robust construction of the neighborhood graph using an adaptive number of neighbors based on the local density of data points is also under consideration.

References

[1] G. Agin and T. Binford. Computer description of curved objects. In *Third International Conferemce on Artificial Intelligence*, 1973.

[2] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Computer Graphics (SIGGRAPH '98)*, 1998.

[3] D. Attali. Squelettes et graphes de Voronoi 2D et 3D. Thèse de doctorat en sciences, Université Joseph Fourier - Grenoble 1, Oct. 1995.

[4] D. Attali. r-regular shape reconstruction from unorganized points. In *ACM Symposium on Computational Geometry*, pages 248–253, Nice, France, 1997.

[5] D. Attali and A. Montanvert. Computing and simplifying 2D and 3D semicontinuous skeletons of 2D and 3D shapes. *Computer Vision and Image Understanding*, 67(3):261–273, Sept. 1997.

[6] E. Bittar, N. Tsingos, and M. Gascuel. Automatic reconstruction of unstructured 3D data: Combining a medial axis and implicit surfaces. In *Eurographics'95*, pages C–457–C–468, Maastrich, The Netherlands, Sept. 1995.

[7] J. Bloomenthal and K. Shoemake. Convolution surfaces. In *Computer Graphics (SIGGRAPH '91)*, volume 25, pages 251–256, July 1991.

[8] J. Bloomenthal and B. Wyvill. Interactive techniques for implicit modeling. *Computer Graphics (1990 Symposium on Interactive 3D Graphics)*, 24(2):109–116, Mar. 1990.

[9] H. Blum. Biological shape and visual science (part I). *Journal of Theoretical Biology*, 38:205–287, 1973.

[10] J.-D. Boissonnat. Shape reconstruction from planar cross sections. *Computer Vision, Graphics and Image Processing*, 44, 1988.

[11] V. Burdin, C. Roux, C. Lefèvre, and E. Stindel. Modeling and analysis of 3-D elongated shapes with application to long bone morphometry. *IEEE Transactions on Medical Imaging*, 15(1):79–91, Feb. 1996.

[12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill, New York, NY, 1990.

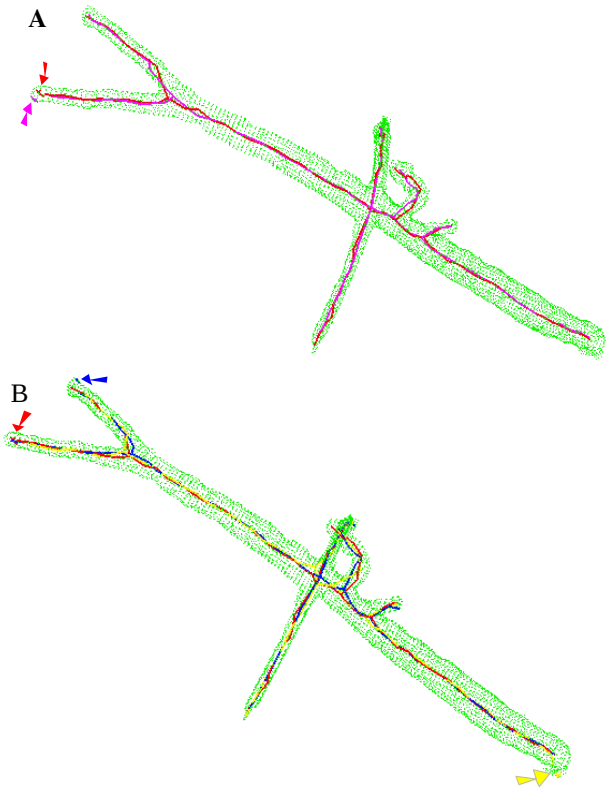

Figure 9. A: two skeletal curves associated to two neighbor points; B: three skeletal curves associated to three different source points.

[13] P. Eilers. Smoothing and interpolation with finite differences. In *Graphics Gems IV*. Academic Press, 1994.

[14] E. Ferley, M.-P. Cani-Gascuel, and D. Attali. Skeletal reconstruction of branching shapes. In *Implicit Surfaces'96*, pages 127–142, Eindhoven, Oct. 1996 (also in Computer Graphics Forum, vol. 16, no. 5, 1997). Eurographics.

[15] J. Goldak, X. Yu, A. Knight, and L. Dong. Constructing discrete medial axis of 3D objects. *International Journal of Computational Geometry and Applications*, 1(3):327–339, 1991.

[16] J. Hart. Morse theory for implicit surface modeling. In *Visualization and Mathematics '97*, Berlin-Dahlem, Sept. 1997.

[17] Y. Kergosien. Topology and visualization: from generic singularities to combinatorial shape modelling. In Y. S. T.L. Kunii, editor, *International workshop on modern geometric computing for visualization*, Tokyo, Japan, 1992. Springer.

[18] F. Lazarus. Courbes, cylindres et métamorphoses pour l'image de synthèse. Thèse de doctorat en sciences, Université PARIS VII, Dec. 1995. http://www-sic.univ-poitiers.fr/ lazarus/publications.html#these.

[19] F. Lazarus and A. Verroust. Décomposition cylindrique de polyèdre et courbe squelette. *Revue Internationale de CFAO et d'Infographie*, 11(4), Nov. 1996.

[20] F. Lazarus and A. Verroust. Level set diagrams of polyhedral objets. Rapport de Recherche INRIA no 3546, Nov. 1998.

[21] R. Mencl and H. Müller. Graph-based surface reconstruction using structures in scattered points sets. In *Computer Graphics International'98*, pages 298–311, June 1998.

[22] J. Mitchell, D. Mount, and C. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16:647–668, 1987.

[23] M. Näf, O. Kübler, R. Kikinis, M. Shenton, and E. Székely. Characterisation of 3D organ shapes in medical image analysis using skeletonization. In *Workshop on Mathematical Methods on Biomedical Image Analysis*. IEEE PAMI, June 1996.

[24] B. Nazarian, C. Chédot, and J. Sequeira. Interactivity and Delaunay triangulation for the reconstruction of tubular anatomical structures. In *16th annual conference of IEEE/Engineering in Medecine and Biology Society*, Nov. 1994.

[25] B. Nazarian, C. Chédot, and J. Sequeira. Automatic reconstruction of irregular tubular structures using generalized cylinders. *MICAD'96 - Revue Internationale de CFAO et d'Infographie*, 11(1-2), 1996.

[26] K. Opitz and H. Pottmann. Computing shortest paths on polyhedra: applications in geometric modeling and scientific visualization. *International Journal of Computational Geometry and Applications*, 4(2), 1994.

[27] C. Pisupati, L. Wolf, W. Mitzner, and E. Zerhouni. A central axis algorithm for 3D bronchial tree structure. In *International Symposium on Computer Vision*, pages 259–264. IEEE PAMI, Nov. 1995.

[28] P. Schneider. An algorithm for automatically fitting digitized curves. In *Graphics Gems*. Academic Press, 1990.

[29] J. Sequeira, R. Ebel, and F. Schmitt. Three-dimensional modeling of tree-like anatomical structures. *Computerized Medical Imaging and Graphics*, 17(4), Nov. 1993.

[30] U. Shani and D. Ballard. Splines as embeddings for generalized cylinders. *Computer Vision, Graphics and Image Processing*, 27(2), Aug. 1984.

[31] Y. Shinagawa, T. Kunii, and Y. L. Kergosien. Surface coding based on Morse theory. *IEEE Computer Graphics and Applications*, 11(5):66–78, Sept. 1991.

[32] B. Soroka, R. Andersson, and R. Bajcsy. Generalized cylinders from local aggregation of sections. *Pattern Recognition*, 13(5), 1981.

[33] S. Takahashi, T. Ikeda, Y. Shinagawa, T. Kunii, and M. Ueda. Algorithms for extracting correct critical points and constructing topological graphs from discrete geographical elevation data. In *Eurographics'95*, pages C–181–C–192, Maastrich, The Netherlands, Sept. 1995.

[34] J. P. Williams and L. B. Wolff. Form from function: a vector field based approach to the analysis of ct images of the vascular tree. In *Workshop on Physic-based Modeling in Computer Vision*, Cambridge, MA, June 1995. IEEE Computer Society Press.

[35] M. Zerroug and R. Nevatia. Volumetric descriptions from a single intensity image. *International Journal of Computer Vision*, 20(1/2):11–42, Oct. 1996.