RECONSTRUCTION AND NAVIGATION OF CYLINDRICAL OBJECTS FROM MEDICAL IMAGES

Yoo-Joo Choi

Dept. of Computer Science and Engineering, Ewha Womans University
11-1 Daehyundong Seodaemungu
Seoul, Korea 120-750, Korea

KyungHa Min

Dept. of Media Technology
Graduate School of Media Communications, Sogang Uni.,
1, Shinsoo-dong, Mapo-gu,
Seoul, Korea

Myoung-Hee Kim

Dept. of Computer Science and Engineering, Ewha Womans University
11-1 Daehyundong Seodaemungu
Seoul, Korea 120-750, Korea

ABSTRACT

This paper proposes a new contour detection method and adaptive reconstruction scheme for the cylindrical organs, such as blood vessels or arteries. Furthermore, we present java-based navigation controller which has been built to examine the inside of cylindrical objects. In the preprocessing procedure, a few preprocessing image filters are applied in order to remove unwanted artifacts from the medical images and to estimate threshold values for the object of interest. We define a context-free grammar, which is proper for properties of contours of cylindrical objects . In the next procedure, we extract contours using advanced radial gradient method and represent contours as context-free grammar derivation trees. We build polygons between two contours efficiently by traversing the derivations trees of the contours. We fly through the reconstructed virtual models using java-based navigation controller and VRML viewer.

1 INTRODUCTION

The virtual endoscopy has emerged as a safe method to investigate the inside of the human organs by the non-invasive ways. Virtual endoscopy has received attention for many parts of the body[1]. The approaches to build the virtual endoscopy are classified into two categories. The first category includes the volume-rendering based approaches and the second one includes the surface model based approaches. The volume-rendering based approaches have an advantage to represent the minute parts of organs, such as the end of bronchus, while on the other side these usually need a lots of performance time. The surface model based approaches support the efficient renderingspeed without an expensive hardware graphics accelerator, however these have the limits to the detail representation of minute parts of organs.

This paper presents an contour-based reconstruction scheme of 3D polygonal surface of the cylindrical object in order to support the effective surface-based virtual endoscopy. For the contour detection, we should apply various preprocessing image filters in order to remove unwanted artifacts and to estimate the threshold values for the objects of interest. We extract contours using modified radial gradients method using estimated threshold values. We radiate a set of rays from the inside of the object of interest, and samples points on the boundary of the object to extract by computing the intersections between the rays and the boundary. The sampled points are connected to a contour using chain code algorithm. The edges on the contours are classified into six types according to the rays that determine the endpoints of the edges, and a contour is represented using context-free grammar. For the reconstruction, we build polygons between two contours efficiently by traversing the derivations trees of the contours. After the initial reconstruction, we can adaptively refine the polygonal surface by refining the contours and modifying the polygonal surfaces according to the refined contours. The resulted surface model is represented by VRML node, and we navigate inside of the reconstructed object using Java-navigation controller, which directly controls the parameters of the VRML node.

The remaining chapters of this paper are as follows. In chapter 2, we briefly review the researches on contour detection and visualization for the volume data, and contour-based reconstruction schemes. In chapter 3, we discuss contour detection algorithm based on the advanced radial gradient method. We present a effective tiling algorithm in chapter 4, and an adaptive refinement scheme in chapter 5. In chapter 6, we discuss the result of implementation and navigation using Java-based navigation controller. Finally, we suggest the conclusion and future works in chapter 7.

2 RELATED WORKS

The approaches for the separation of image into regions that are meaningful for a specific task, called image segmentation, can be classified into one of three groups[2]: region-based, edge-based, or classification. Typically, region-based and edge-based segmentation techniques exploit, respectively, within-region similarities and between-region differences, whereas a classification technique assigns class labels to individual pixels or voxels based on feature values, such as pixel intensities or gradient magnitudes. In edge-based segmentation approaches, the processed image is described in terms of the edges (contours) between different regions. Edges can be detected with a variety of edge operators. The most popular ones are the Marr-Hildreth or Log(Laplacian-of Gaussian), Sobel, Prewitt, and Canny operators. These approaches using edge operators represent contours as a set of points and do not permit detail-level of contour shape to be controlled.

Keppel [13] originally proposed a reconstruction scheme for a surface from 2D contours. He built toroidal graph between the points on the contours and proposed a search scheme on the graph to generate polygons from the points. Fuchs et al.[10] improved the Keppel's method by applying divide-and-conquer approach for searching toroidal graph. Christiansen et al. [7] computed a midpoint between two contours on the same level to unite the two contours into one. To solve the tiling problem, they introduced greedy search algorithm for the toroidal graph proposed by Keppel. Boissonnat [4] computed tetrahedralization of the pace between two contours using Delaunay triangulation of a pair of successive contours. The internal edges of the tetrahedralization are removed to obtain a solid interpolation of two contours. Ekoule et al. [9] decomposed non-convex contours into a set of convex sub-contours and applied tiling algorithm to tile two very dissimilar contours. Meyers et al. [17] approximated the contours to tile using ellipses and constructed minimum spanning tree for efficient tiling. Jones et al. [12] approximated contours on the slices using discrete field functions and tiled the contours by interpolating the functions. Bajaj et al. [3] attacked the three problems simultaneously by imposing a set of three constraints on the reconstructed surface and then deriving precise correspondence and tiling rules from these constraints. From these rules, they provided a tile scheme that can process contours of arbitrary topology. Oliva et al. [19] projected contours on a common plane and applied generalized Voronoi diagram on the area of intersection. Two contours are tiled by interpolating the triangles of the Voronoi diagram. Recently, Cong et al. [8] constructed partial differential equations whose solution is a field function that describes a contour using equally important criterion and tiled the contours by solving the PDE, and Klein et al. [14] computed medial axis in the area of intersection of two contours, which are projected onto a common plane and found correspondence between the points on the contours. Marsan et al. [16] applied morphological dilation operators to compute the interpolation of the contours, and Fujimura [11] proposed an isotopic deformation scheme to interpolate the contours for tiling .

3 CONTOUR DETECTION

3.1 preprocessing

At the first step of preprocessing, we designate the region of interest (ROI) from the images. Therefore, we concentrate the preprocessing procedures only for the pixels inside the ROI to improve the performance. In this research, we apply median filter and gaussian filter in order to minimize the effects of artifacts of ultrasonic images. Median filter that closes gab with preserving edge sharpness reduces speckles, and gaussian filter that blurs the image reduces noises. Furthermore, we apply morphological dilation operation and morphological erosion to remove small gaps in order. In order to estimate the threshold values of object of interest, we select 4 or more points of inside of interested object in the first and last images. For other images, 4 points of inside of interested object are automatically selected based on selected points in the first and last images. And we compute the mean ϖ and standard variance σ of the pixel values of points of the cardinal spline curve, which are determined by 4 or more selected points. The range of pixel values of interested object is fixed for $\varpi \pm \sigma$, which is used as threshold values for contour extraction.

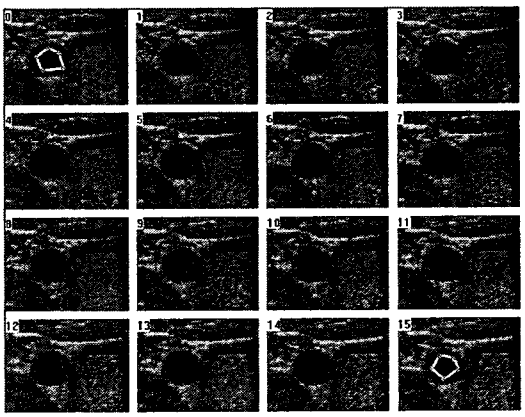


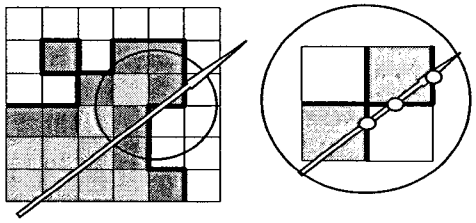
Figure 1. Initial selected points

3.2 Extraction of contours

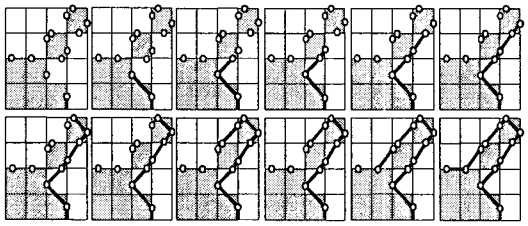
For each sampled slice of volume data, we extract a contour of the object on the slice using modified radial gradient method. The conventional radial gradient method [5, 6, 24] constructs a contour of an object by sampling pixels on the boundary of the object and connecting them to a closed simple polygon. To sample the pixels, they radiate a set of rays from a center pixel, and find the intersection pixels between the ray and the boundary of the object.

We improve the conventional radial gradient method in the following features to extract more accurate contours.

1. For each intersection pixel, we compute intersection points between the ray and the borders of the intersection pixel (darker gray pixels in Figure 2 (a)). Note that a contour is constructed by connecting the intersection points, not the intersection pixels. Figure 2 (a) illustrates the intersection pixels and their intersection points.
2. To connect the intersection points, we design a modified chain code algorithm [20] that traverses the border edges of the boundary pixels of the closed region. Figure 2 (b) indicates example to connect intersection points using modified chain code algorithm.



(a) Intersecting pixels and their intersection points



(b) Example of connecting intersection points

Figure 2. Modified Radial Gradient Method

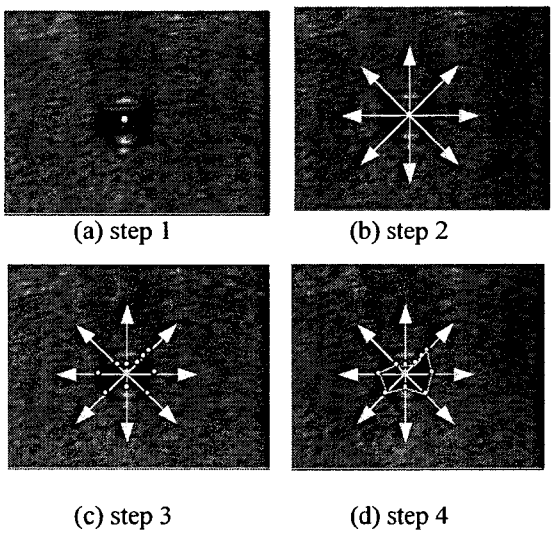
The proposed modified radial gradient method extracts contours in the following steps.

Step 1. A center pixel is chosen inside the object to extract.

Step 2. A set of rays, denoted as $r_0, r_1, \ldots, r_{m-1}$, where m denotes the number of rays, is radiated from the center pixel in counterclockwise order. For the coherence of the contours on the slices, r_0 is cast in the direction of $(1,0,0)$.

Step 3. The intersection points on the intersection pixels are computed from the rays and the boundary of the ROI.

Step 4. We connect the intersection points by traversing the boundary of the object using the modified chain code algorithm. Figure 3 illustrates the four steps of the modified radial gradient method to extract a contour from an ROI.



(a) step 1 (b) step 2

(c) step 3 (d) step 4

Figure 3. Four Steps of Modified Radial Gradient Method

A contour C of a object on the slice is defined as a closed simple polygon: $C = \{v_0, v_1, \ldots, v_{n-1}\}$, where n is the number of intersection points on the object. Note that for v_i and v_j, where $v_i \in C$ and $v_j \in C$, $i > j$ indicates that v_i is visited before v_j when we traverse the boundary of the ROI using the modified chain code algorithm. We classify the edges on the contour into six types according to the rays that determine the endpoints of the edges.

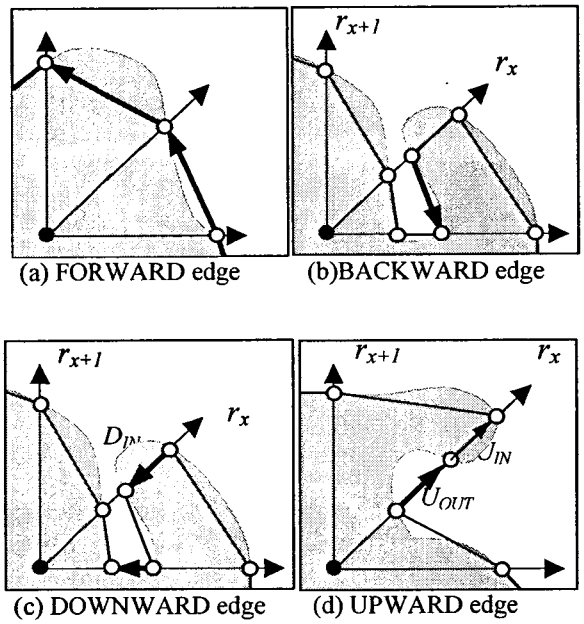
Definition 1.

An edge, e.g. (v_i, v_{i+1}), on the contour is classifies into six types: $\{FORWARD, BACKWARD, UP_{IN}, UP_{OUT}, DOWN_{IN}, DOWN_{OUT}\}$.

1. An edge (v_i, v_{i+1}) is a *FORWARD* edge (abbreviated as *F*), if the rays that determine v_i and v_{i+1} are r_x and r_{x+1}, respectively.
2. An edge (v_i, v_{i+1}) is a *BACKWARD* edge (abbreviated as *B*), if the rays that determine v_i and v_{i+1} are r_{x+1} and r_x, respectively.
3. An edge (v_i, v_{i+1}) is UP_{IN} edge (abbreviated as U_I), if the rays that determine v_i and v_{i+1} are r_x (identical), and the direction from v_i to v_{i+1} is along r_x, and the edge lies inside the ROI.

4. An edge (v_i, v_{i+1}) is UP_{OUT} edge (abbreviated as U_O), if the rays that determine v_i and v_{i+1} are r_x (identical), and the direction from v_i to v_{i+1} is along r_x, and the edge lies outside the ROI.
5. An edge (v_i, v_{i+1}) is $DOWN_{IN}$ edge (abbreviated as D_I), if the rays that determine v_i and v_{i+1} are r_x (identical), and the direction from v_i to v_{i+1} is reverse to r_x, and the edge lies inside the ROI.
6. An edge (v_i, v_{i+1}) is $DOWN_{OUT}$ edge (abbreviated as D_O), if the rays that determine v_i and v_{i+1} are r_x (identical), and the direction from v_i to v_{i+1} is reverse to r_x, and the edge lies outside the ROI.

The six types of the edges are illustrated in the Figure 4.



(a) FORWARD edge (b)BACKWARD edge

(c) DOWNWARD edge (d) UPWARD edge

Figure 4. Six types of the edges on a contour

Lemma 1.

If edges on a contour are replaced by the symbols according to the Definition 2, we can represent the contour as a string that is recognized by the context-free grammar $G = \{V_N, V_T, P, S\}$, where

$V_N = \{F, B, U, U_R, U_I, U_O, D, D_R, D_I, D_O\}$,
$V_T = \{f, b, u_i, u_o, d_i, d_o\}$,
$P = \{P_1: S \rightarrow F \ldots F,$
$P_2: F \rightarrow D\,F\,U \mid D\,F \mid F\,U \mid f,$
$P_3: B \rightarrow b,$
$P_4: D \rightarrow D\,D \mid D\,U \mid U\,D \mid D_I\,D_O,$
$P_5: D_I \rightarrow F\,D_I\,B \mid D_I\,U_R \mid d_i,$
$P_6: D_O \rightarrow B\,D_O\,F \mid D_O\,U \mid d_o,$
$P_7: D_R \rightarrow D_O\,D_I,$
$P_8: U \rightarrow U\,U \mid U\,D \mid D\,U \mid U_O\,U_I,$
$P_9: UI \rightarrow B\,U_I\,F \mid U_I\,D \mid u_i,$
$P_{10}: UO \rightarrow F\,U_O\,B \mid U_O\,D_R \mid u_o,$
$P_{11}: U_R \rightarrow U_I\,U_O\}$,
$S = \{S\}$.

We denote that the symbols $\{U, U_R, U_I, U_O\}$ and $\{D, D_R, D_I, D_O\}$ are homogeneously symbols, respectively, since each set of symbols represents edges of same direction.

4 MESH CONSTRUCTION

To build a polygonal surface between two contours, we find geometrically matching edges or vertices between two contours. In this paper, we exploit the derivation tree of the context-free grammar G in determining the matching pairs. This approach improves the computational time required in the tiling stage from the fact that the corresponding pairs are determined without geometric computations.

Definition 2.

The derivation tree T of G is composed of the following four types of nodes:

1. **Root node** stores S, the start symbol of G, and has m level-1 child nodes, where m denotes the number of radiated rays.
2. **Level-1 internal node** stores a nonterminal symbol F and a FORWARD edge.
3. **Internal node below level-1** stores a nonterminal symbol of G.
4. **Leaf node** stores one of six terminal symbols of G and the edge on the contour.

We apply the conventional parsing algorithm of context-free grammar to build the derivation tree of a contour. During the parsing procedure, we scan the string from left to right, and build the derivation tree according to a bottom-up scheme as follows:

1. If we apply one of the production rules of G to a terminal symbol and derive a nonterminal symbol, a leaf node that stores the terminal symbol is created.
2. If one of the production rules of G is applied to some nonterminal symbols to derive a new nonterminal symbol, we build an internal node that stores the nonterminal of the left-hand side of the production rules. The internal node has child nodes that store the symbols in the right-hand side of P.
3. Finally, we build a root node that stores start symbol S of G. The root node has m level-1 child nodes that store FORWARD edges.

Figure 5. illustrates a parsing tree for an exampled contour

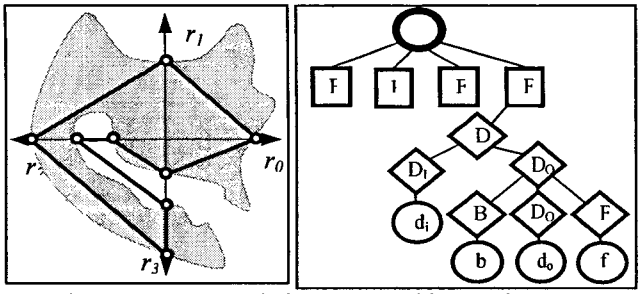
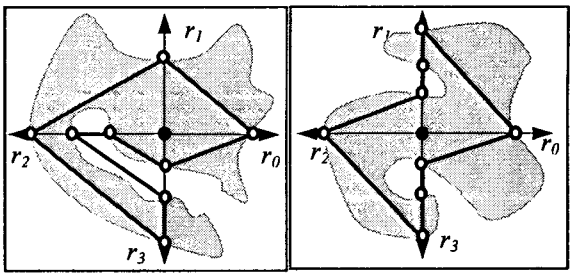
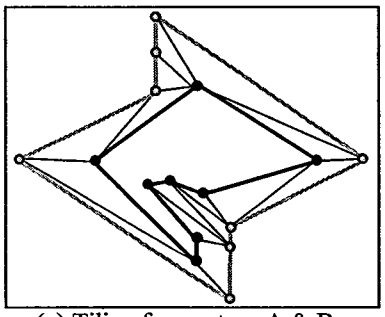


Figure 5. An exampled contour and its parsing tree



(a) Contour A (b) Contour B



(c) Tiling for contour A & B

Figure 6. An exampled tiling of two contours

We find the geometrically matching edges or vertices on two contours through the simultaneous traversal of the derivation trees. The generated polygon is a triangle determined by an edge on a contour and its corresponding vertex on the neighboring contour. Note that a triangle is determined by an edge and its opposite vertex. Therefore, a pair of an edge and its matching vertex creates one triangle, and a pair of two matching edges creates two triangles. The algorithm for finding geometrically matching pairs and generating polygons is as follows:

1. At root nodes, we visit the level-1 nodes of both trees from left child to right child simultaneously.
2. At the *i*-th level-1 internal nodes on both trees, the *F* edges stored in the nodes are the corresponding edges to each other. We traverse the child nodes that store the same symbols simultaneously
3. At internal nodes below level-1, we simultaneously traverse the child nodes that store the identical symbol or the homogeneous symbols with the symbol in the visiting node. The edges stored in the unvisited nodes have corresponding vertices instead of corresponding edges. In such cases, a triangle is generated for the pair of an edge and its matching vertex.
4. At leaf nodes, the edges stored in the leaf nodes are the corresponding edges to each other. Therefore, we generate two triangles from the leaf nodes.

5 ADAPTIVE REFINEMENT

In the previous chapter, we described a procedure for building a polygonal surface from 3D ultrasound volume dataset. In this chapter, we present an algorithm that refines the polygonal surface adaptively. The refinement scheme is composed of two phases: refinement of the contour, and the refinement of the polygonal surface.

5.1 Refinement of the contour

To refine a contour, we cast more rays between the existing rays on the slice and compute new intersection points. The production rules P_2 and P_3 of G are modified to P_2' and P_3', respectively, to recognize the refined contour as follows:

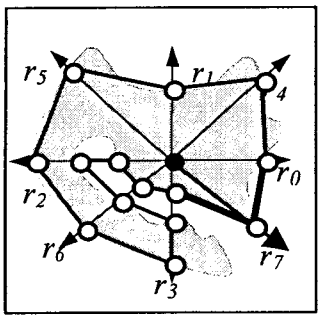
P_2': $F \rightarrow FF \mid DFU \mid DF \mid FU \mid f$

P_3': $B \rightarrow BB \mid b$

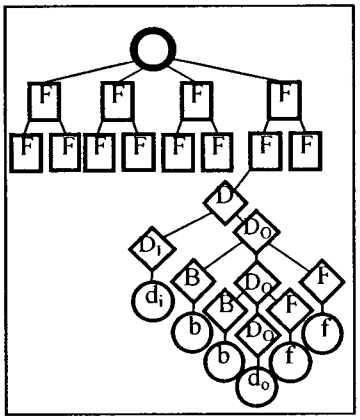
The procedure to modify the parsing tree for the refined contour is as follows:

1. We substitute the terminal symbols of the refined edges by non-terminal symbols in the string that represent the contour.
2. The leaf nodes of the parsing tree that store the terminal symbols are removed.
3. The nonterminals in the modified string are derived to the terminals according to the modified grammar.
4. The parsing tree is modified according to the derivation of the nonterminals in the modified string.

Figure 7 illustrates a refinement of a contour illustrated in Figure 6 and its modified parsing tree

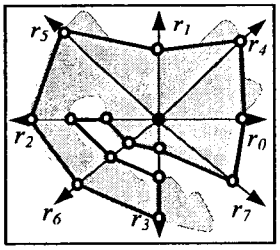


(a) Refinement of a contour byt casting four more Rays : { r4, r5, r6, r7 }

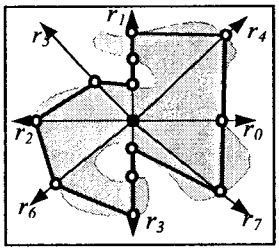


(b) Modificaiton of the parsing tree

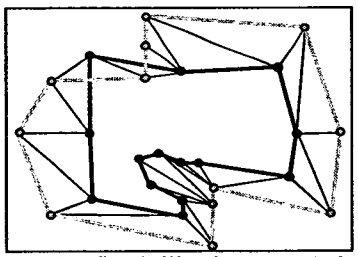
Figure 7. Refinement of a contour and its parsing tree



(a) Refined contour A



(b) Refined contour B



(c)Refined tiling between A & B

Figure 8. Refinement of tiling by casting more rays

5.2 Refinement of the polygonal surface

The refinement of the edges on the contour is implemented through modifying the derivation tree of the contour. Note that an edge on the contour belongs to a triangle on the polygonal mesh. Therefore, if an edge is refined into new edges, the triangle determined by the edge is also refined into triangles containing the new edges. We refine the polygonal surface through traversing the modified parsing trees, and generate polygons according to the scheme presented in Section 4. Figure 8 illustrates the refinement of the tiling suggested in Figure 6.

6 IMPLEMENTATION & RESULTS

The proposed algorithm is implemented on a PC platform with 800 MHz Pentium III CUP and 256 MB Main Memories. The software environment is Visual C++ 6.0 with OpenGL library. We tested the proposed scheme for the reconstruction of two pipe-typed objects: a phantom object and an carotid artery of the neck. Figure 9 illustrates four levels of details for the artery object. The important advantage of the proposed algorithm is to preserve the features of contours. The existed algorithms depend on the geometry information of the contours, so the existed tiling procedures are difficult to preserve features of the contours. The object in Figure 10 is a twisted object which is made by screwing and moving a slice object. The proposed method casts the rays in the each slice image with rotation of rays. Figure 10 compares the results of reconstruction using toroidal search method and the proposed method. Using the proposed method, the reconstructed object preserve the features of the shape of original object.

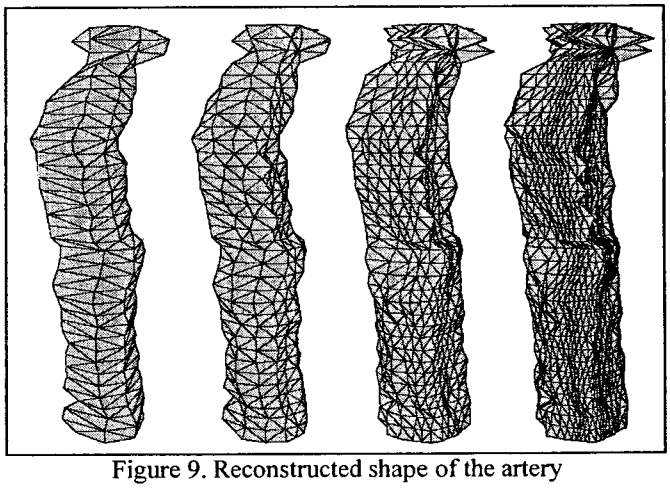
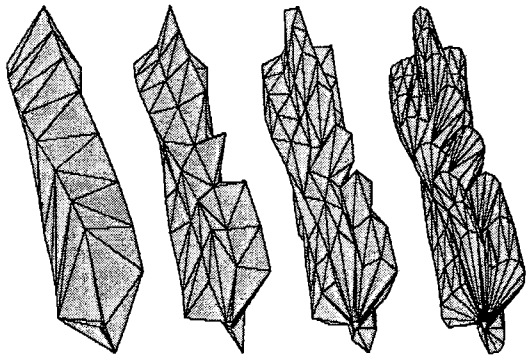
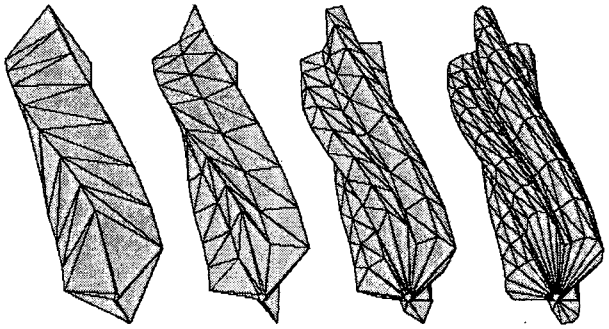


Figure 9. Reconstructed shape of the artery



(a) reconstructed shape using toroidal search method



(b) reconstructed shape using the proposed method

Figure 10. Reconstructed shape of twisted object

In order to investigate the inside of the cylindrical object, we developed simple java-based navigation system. This system supports two views for inside and outside shape of the object and location indicator. Figure 11 shows a execution sample of java-based navigation system.

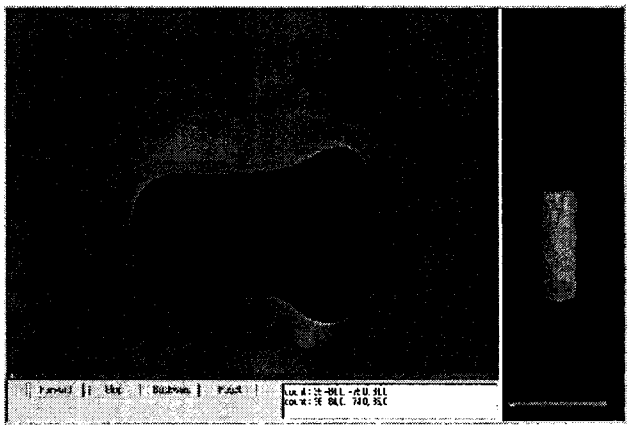


Figure 11. Java-based navigation controller

7 CONCLUSION & FUTURE WORKS

In this paper, we proposed a new scheme that adaptively reconstructs pipe-typed human organs from the volume data acquired through 3D ultrasonic devices. The contributions of this paper is as follows:

- We proposed a modified radial gradient method that extracts an exact contour from an image slice in an adaptive way.
- We exploited a context-free grammar that generates polygons between two contours in a very efficient way. Since we find the geometrically matching edges or vertices between two contours by the traversal of the trees, the tiling procedure takes little computational costs.
- From the modified radial gradient scheme that extracts the contours in an adaptive way, the polygonal surface that approximates the pipe-typed human organ can be generated in an adaptive way.

We are going to solve the branching problems for the future work to reconstruct the bifurcated pipe-typed human organs. For this purpose, the context-free grammar that represents a single contour will be extended to represent multiple contours. Based on the extended context-free grammar, the geometrically corresponding vertices and edges between more than two contours are determined to generate polygons for the branched contours. As an application of the results, a virtual environment that navigates inside of the pipe-typed human organ in real-time is going to be developed.

ACKNOWLEGEMENTS

This work was supported in part by the Korean Ministry of Information and Communication under the Information Technology Research Center (ITRC) Program and in part by the Korean Ministry of Science and Technology under the National Research laboratory(NRL) program.

REFERENCES

[1] Milan Sonka, J. Micahel Fitzpartrick. 2000. *Handbook of Medical Imaging, Volume 2. Medical Image Processing and Analysis.* SPIE PRESS.

[2] K. S. Fu and J. K. Mui, "A survey on image segmentation," Pattern Recognition, vol. 13, pp. 3-16, 1981.

[3] Bajaj, C. L., Coyle, E. J., and Lin, K. N., "Arbitrary Topology shape reconstruction from planar cross sections," Graphical Models and Image Processing, Vol. 58, No. 6, pp. 524-543, 1996.

[4] Boissonnat, J. D., "Shape reconstruction from planar cross sections," Computer Vision, Graphics, and Image Processing, Vol. 44, pp. 1-29, 1988.

[5] Buda A. J., Delp E. J., Meyer C. R., Jenkins J. M., Smith D. N., Bookstein F. L., and Pitt B., "Automatic computer processing of digital 2-dimensional echocardiograms," Am J Cardiol 52:384, 1983
[6] Garcia E, Gueret P, Bennett M, Corday E, Zwehl W, Meerbaum S, Corday S, Swan HJC, and Berman D, "Real-time computerization of two-dimensional echocardiography," Am Heart J 101:783, 1981
[7] Christiansen, H. N. and Sederberg, T. W., "Conversion of complex contour line definitions into polygonal element mosaics," Proceedings of SIGGRAPH 78, pp. 187-192, 1978.
[8] Cong, G. and Parvin, B., "An algebraic solution to surface recovery from cross-sectional contours," Graphical Models and Image Processing, Vol. 61, No. 4, pp. 222-243, 1999.
[9] Ekoule, A. B., Peyrin, F. C., and Odet, C., L., "A triangulation algorithm from arbitrary shaped multiple planar contours," ACM Transactions on Graphics, Vol. 10, No. 2, pp. 182-199, 1991.
[10] Fuchs, H., Kedem, Z. M., and Uselton, S. P., "Optimal surface reconstruction from planar contours," Communications of ACM, Vol. 20, No. 10, pp. 693-702, 1977.
[11] Fujimura, K., "Shape reconstruction from contours using isotopic deformation," Models and Image Processing, Vol. 61, No. 3, pp. 127-147, 1999.
[12] Jones, M. W. and Chen, M., "A new approach to the construction of surfaces from contour data," Computer Graphics Forum, Vol. 13, No. 3, pp. 75-84, 1994.
[13] Keppel, E., "Approximating complex surfaces by triangulation of contour lines," IBM Journal of Res. Dev. Vol. 19, pp. 2-11, 1975.
[14] Klein, R., Schilling, A., and Strasser, W., "Reconstruction and simplification of surfaces from contours," Proceedings of Pacific Graphics '99, pp. 198-207, 1999.
[15] Marques, J. S., "Alignment-by-Reconstruction for 3D ultrasound Imaging", Proceedings of the International Conference on Pattern Recognition (ICPR'00), 2000.
[16] Marsan, A. L. and Dutta, D., "techniques for automatically tiling and skinning branched objects," Computers \& Graphics, Vol. 23, No. 1, pp. 111-126, 1999.
[17] Meyers, D., Skinner, S., and Sloan, K., "Surfaces from contours," ACM Transactions on Graphics, Vol. 11, No. 3, pp. 228-258, 1992.
[18] Min, K.H., and Lee, I.K., "Multiresolutional Reconstruction of Polygonal Surfaces from Contours using Context-Free Grammar," submitted.
[19] Oliva, J. M., Perrin, M., and Coquillart, S., "3D reconstruction of complex polyhedral shapes from contours using a simplified generalized Voronoi diagram," Computer Graphics Forum, Vol. 15, No. 2, pp. 397-408, 1996.
[20] Pitas, I., Digital Image Processing Algorithms, Prentice Hall, 1993.
[21] Sakas, G., Schreyer, L.-A., and Grimm, M., "Preprocssing and Volume Rendering of 3D Ultrasonic Data", IEEE Computer Graphics and Applications, Special Issue, vol. 15, No.4, July 1995.
[22] Sanches, J. M., Marques, J. S., "A Rayleigh reconstruction / interpolation algorithm for 3D ultrasound", Pattern Recognition Letters, Volume 21, Issue 10, September 2000.
[23] Sonka, M., Fitzpatrick, J. M., "Handbook of Medical Imaging volume 2. Medical Image Processing and Analysis", SPIE Press, 2000.
[24] Storton, D. J., and Collins, S. M., "Digital Signal and Image Processing in Echocardiography," Heart J, 110:1266-1283, 1985.

AUTHOR BIOGRAPHIES

Yoo-Joo Choi is a Ph.D student in the Department of Computer Science and Engineering at Ewha Womans University of Korea. She received her BS and MS in Computer Science from the Ewha Women's University, Seoul, South Korea, in 1989 and 1991. Her research interests are in medical image processing, computer graphics, and virtual reality. Her email address is choirina@mm.ewha.ac.kr.

KyungHa Min is a research faculty in the Department of Media Technology at Graduate School of Media Communications of Sogang University. He received his BS from KAIST, Taejon, Korea in 1992, and his MS and Ph.D from POSTECH, Pohang, Korea in 1994 and 2000, respectively. His research interests are computer graphics and media technology. His email address is minkh@sogang.ac.kr

Myoung-Hee Kim has been a professor in the Department of Computer Science and Engineering at Ewha Womans University of Korea since 1987. She received her MS degree from Seoul National University of Korea in 1979 and her PhD degree from Universitaet Goettingen of Germany in 1986. Her current research interest include visualization, simulation and virtual reality. Her email address is mhkim@mm.ewha.ac.kr