# Three-dimensional interactive virtual endoscopy

Guangming Li, Jie Tian[*], Mingchang Zhao and Huiguang He
*Medical Image Processing Group, Institute of Automation, Chinese Academy of Sciences, P.O. Box 2728, Beijing, 100080, P.R. China*

**Abstract.** Virtual endoscopy is a new development of diagnostic imaging, covering a wide range of potential clinical applications. This paper introduced a system framework for virtual endoscopy. Our research efforts concentrated on mesh smoothing, automated centerline extraction and view-dependent level-of-detail rendering techniques. First, the fairing surface was obtained by smoothing the normal vectors of the reconstructed model. Then, the modified reliable path method was used to generate the accurate skeleton. After that, the view-dependent level-of-detail rendering techniques, including mesh simplification, mesh parameterization and mesh subdivision, were used to achieve the purpose of real time rendering. Experimental results for the images of cerebral blood vessels, knees, and trachea using our solution were given.

Keywords: Virtual endoscopy, mesh smoothing, path planning, mesh simplification, view-dependent subdivision, virtual navigation

## 1. Introduction

Virtual endoscopy is an integration of medical imaging, computer graphics and virtual reality techniques. Compared to traditional endoscopy, it is noninvasive, cost-effective, highly accurate, free of risks, and easily tolerated by the patient. Virtual endoscopy allows physicians to navigate inside human organs to detect abnormalities. It has been under development in many clinical areas, such as virtual colonoscopy and virtual angioscopy.

A meaningful virtual endoscopy system touches on many techniques in medical imaging and virtual reality. Figure 1 shows the global structure of virtual endoscopy. It comprises three primary components: centerline extraction, fairing surface generation and real time rendering.

The purpose of centerline extraction is to define the path that guides the movement of the camera during navigation. A good path should stay near the center of the model as much as possible and should be one-voxel-wide without self-intersection. Furthermore, the path planning procedure should be fast and automatic [1].

Marching Cubes (MC) algorithm is widely used to reconstruct 3-D surface model in medical image processing. An enhanced MC algorithm – single surface tracking – to reconstruct 3D models [2] is often used.

The normal vectors of reconstructed surface are usually obtained by computing the voxel gradient of the grey-scale images. While there are many cases where the gradient of grey-scale image data provides

[*]Corresponding author: Jie Tian, Senior Member, IEEE. Tel.: +1 8610 62532105; Fax: +1 8610 62527995; E-mail: tian@doctor.com; jie.tian@mail.ia.ac.cn; URL: http://www.3dmed.net.

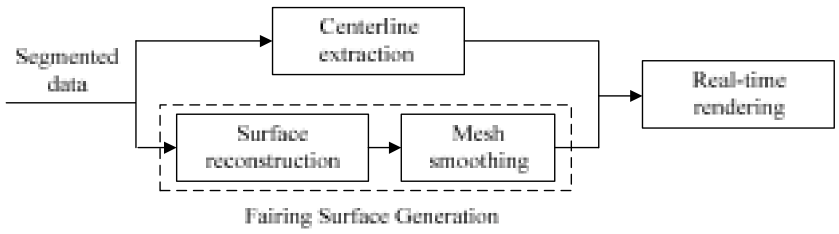


Fig. 1. The schematic of a virtual endoscopy.

good estimates of surface normals, when surfaces are not easily extracted from the grey-scale data, or the original data is just binary images, these estimates can produce severe artifacts [3]. In this case, the estimated normal vectors change dramatically, which results in "terraced surface". In order to remove the stair case artifacts, mesh-smoothing techniques are used. Because only normal vectors are smoothed while the vertices don't move, the features of medical models can be kept.

The most efficient smoothing algorithm is Laplacian smoothing [4]. It removes the noise effectively and rapidly, but at the same time it develops unnatural deformations and produces shrinkage. Although Laplacian smoothing is not suitable for many applications in computer graphics because it changes the model greatly, it is really a good method in virtual endoscopy. The reason is that only normal vectors need to be smoothed and the vertices need not move, thus the shape of the model will not change.

The research work on path planning in recent years can be divided into two categories: topological thinning and distance mapping. The main idea of topological thinning is peeling off the object layer by layer until just the skeleton remains [5,6]. Although topological thinning can provide high quality results, it is computationally expensive. Distance mapping method is developed based on the fact that the centerline of the object should be the point sets that have the maximum distances to the closest surface voxels [1,7–9]. It often uses the Dijkstra's method to generate the centerline. This method is faster than topological thinning, but it cannot preserve the topology of the object.

With the development of acquisition capabilities, the amount of data in medical images becomes larger and larger. A 3-D medical image model often consists of millions of triangles which make real time rendering very difficult. However, representation of highly complex data is not always necessary. For instance, what the camera sees during navigation is usually a small fraction of the entire surface, so we don't need to render all the details of the surface. Level of detail (LOD) scheme is a feasible method to solve this problem. LOD represents the distant objects with a lower level of detail and the nearby objects with a higher level of detail, which can accelerate rendering and increase interactivity. LOD has been used mostly in terrain rendering [13–16]. Among these, the view-dependent progressive mesh (VDPM) is the most important one. However, VDPM algorithm requires a high-end workstation and is time-consuming. Therefore it is not suitable for virtual endoscopy

We developed a new view-dependent level-of-detail algorithm to reduce the number of triangles submitted to the graphics pipeline. Our method consists of three steps: mesh simplification, mesh parameterization and view-dependent mesh subdivision.

In this paper, we systemically describe our research work in virtual endoscopy, including mesh smoothing, path planning, and view-dependent rendering. The mesh smoothing techniques algorithm will be described in Section 2. We will also present an efficient automatic centerline extraction algorithm combining distance transformation and Hessian Matrix in Section 3. We will describe the details of this algorithm of view-dependent level-of-detail in Sections 4. The experimental results of each step are given in the corresponding sections. We concluded our work in Section 5.

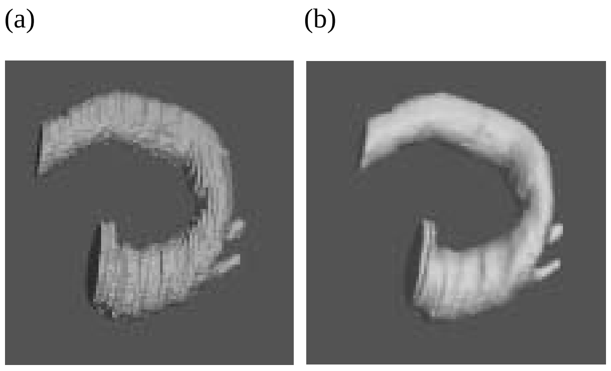


Fig. 2. A small section of abdominal artery. (a) Terraced surface, and (b) Smoothed surface.

## 2. Fairing surface generation

We use mesh-smoothing techniques to remove the stair case artifacts. Here we only smooth the normal vectors while leave the vertices fixed because any movement of vertices will change the shape of the surface model more or less, and also, it is the abrupt changes of normal vectors, not vertices, which produce stair case artifacts.

We smooth every normal vector through its 1-order neighborhood. Let $n_i$ be the normal vector of the vertex $v_i$, then, for $v_i$ and its adjacent vertices $v_j$ ($j \in i^*$, $i^*$ is the star neighborhood of $v_i$), our discrete Lapalacian operator can be expressed as

$$\Delta n_i = \sum_{j \in i^*} w_{ij}(n_j - n_i) \tag{1}$$

where $\sum_{j \in i^*} \cdot w_{ij} = 1$. The weightscan be chosen in many ways. The simplest one is to set the weights equal to each other: $w_{ij} = 1/mi \in i^*$, where $m$ is the valence of $v_i$. Another choice is to set the weights as the inverse distance between $v_i$ and $v_j$: $w_{ij} = \|v_i - v_j\|^{-1}$. When all the displacements of normal vectors are calculated, the normal vectors are updated by the formula

$$n_i^{'} = n_i + \lambda \Delta n_i = (1 + \lambda \Delta) n_i \tag{2}$$

Here $\lambda$ is a small positive scale factor between 0 and 1 which can be used to control the smoothing speed.

Figures 2 and 3 are two examples of our method. In Fig. 2, a small section of abdominal artery is shown; (a) and (b) are surface models before and after smoothing, respectively. Figure 3 shows a reconstructed model for head before and after smoothing. We can see the improvement of image fidelity from these two examples.

## 3. Automated path planning

The extracted path using topological thinning method usually leads to many branches on the path, which will make the fly-through much more difficult to control [7]. Furthermore, the path planning procedure is very slow using topological thinning method. Therefore, as many other researchers have done, we use distance mapping method to extract the path. Our algorithm combines distance mapping

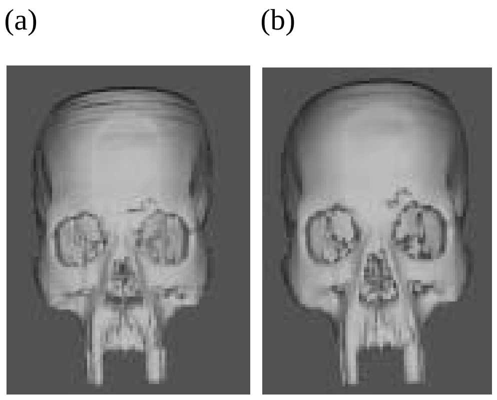


Fig. 3. Head model. (a) Terraced surface, and (b) Smoothed surface.

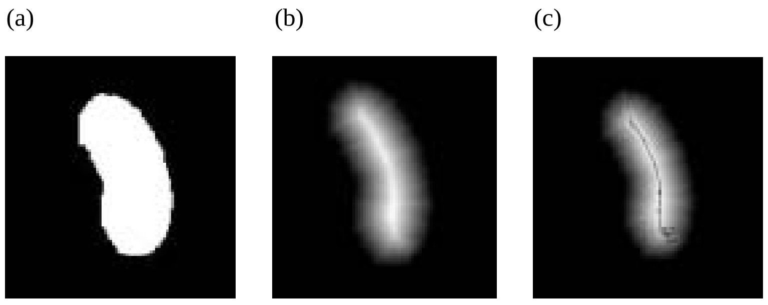


Fig. 4. Ridge extraction using Hessian Matrix. (a) Segmented image, (b) Distance mapping, and (c) Extracted ridge.

with Hessian Matrix. It consists of four steps: distance transformation, ridge extraction using Hessian Matrix, visibility test with adaptive visibility sphere radius and shortest path generation. For distance transformation, we use real Euclidian distances [10]. Figures 4(a) and 4(b) are the binary images after segmentation and its distance mapping.

## *3.1. Ridge extraction using hessian matrix*

A common approach to analyze the local behavior of $nD$ images is to consider its Taylor expansion in the neighborhood of a point $P$ [11]:

$$I(P + \Delta P) \approx I(P) + \Delta P^T \nabla I(P) + \Delta P^T H(P) \Delta P \tag{3}$$

where $\nabla I(P)$ is the gradient vector and $H(P)$ is the Hessian Matrix which is built of the second partial derivatives of $I$. The Hessian Matrix for 3D volumes is a 3*3 symmetric matrix with three real-valued eigenvalues.

$$H(P) = \begin{pmatrix} I_{xx} \; I_{xy} \; I_{xz} \\ I_{yx} \; I_{yy} \; I_{yz} \\ I_{zx} \; I_{zy} \; I_{zz} \end{pmatrix}, \tag{4}$$

where $I_{xy} = \frac{\partial I}{\partial x \partial y}$.

For a point $P$, we order the three eigenvalues of its Hessian Matrix, according to $\lambda_1 < \lambda_2 < \lambda_3$. Let the vector $v_1, v_2$ and $v_3$ associated with $\lambda_1$, $\lambda_2$ and $\lambda_3$. Then two conditions must hold for $P$ to be on a maximum convexity height ridge [12].

First, the first two eigenvectors of Hessian Matrix at $P$ must have negative eigenvalues, i.e. $\lambda_1 \leqslant \lambda_2 < 0$. By this condition, the directions $v_1$ and $v_2$ are assumed to be transverse directions. Second, the projection of the gradient at $P$ onto the transverse directions must be equal to zero, i.e., $v_I \bullet \nabla I = 0, v_2 \bullet \nabla I = 0$.

The red line (central line) in Fig. 4(c) shows the extracted ridge using Hessian Matrix. It represents the shape of the centerline approximately. However, it is obviously seen that this ridge cannot be directly used as the path to guide the movement of the camera during navigation because it has some small branches and isolated points. Visibility test will be performed to remove these useless branches and points. In Section 3.2, we will demonstrated that the Hessian Matrix not only can be used to extract the ridge but also can be employed to adaptively determine the visibility sphere radius.

## *3.2. Visibility test*

The purpose of visibility test is to ensure that each voxel in the surface can be seen by at least one point in the path. A simple visibility test algorithm has been proposed using Reliable Path method [7]. First, at the center of voxel $p$, insert a sphere of radius $r$ representing the range within $p's$ sight. The radius $r$ is usually assigned to be one to three times the value of the maximal voxel-to-surface distance in the model. That is to say, $r = k * d$, $1 < k < 3$. Second, shoot a ray ps from $p's$ center to the center of each surface voxel $s$ within the sphere. If the ray doesn't intersect any other surface voxel, $s$ is visible. Then remove $s$ from boundary voxel list. If none of the boundary voxels that a path point can "see" is in the boundary voxel list, remove this path point from the path list.

The main problem of this method is the determination of the visibility sphere radius. It uses a fixed radius for all path points. Apparently, an adaptive radius, which is determined by the local behavior of the path point, is more reasonable. We use Hessian matrix to achieve this goal. For a point $p$, its visibility sphere radius is determined by the formula $r = k * d$. Although it has the same form as Reliable Path method, it has different meanings. The coefficient k is not simply chosen from one to three, and the variable $d$ is not the maximal voxel-to-surface distance of all the internal points, but the voxel-to-surface distance at voxel $p$.

In fact, the tubular structure often appears elliptical than circularly symmetric. Therefore, we can use the half major axis of the ellipse as the visibility sphere radius. However, from distance transformation, we can only get the minimum voxel-to-surface distance, which can be considered as the half minor axis of the ellipse. Here we use the eigenvalues of Hessian Matrix to find the relationship between the major axis and the minor axis of the ellipse.

For a voxel $p$ in initial path, let $\lambda_1, \lambda_2, and\ \lambda_3$ be the eigenvalues of its Hessian Matrix, such that $\lambda_1 \leqslant \lambda_2 \leqslant \lambda_3$. $v_1$, $v_2$ and $v_3$ are eigenvectors corresponding with $\lambda_1, \lambda_2, and\ \lambda_3$. From Section 3.1, we know that $v_1$ and $v_2$ are transverse directions, so we can estimate the ratio of major axis and minor axis of the ellipse through the ratio of $\lambda_1$ and $\lambda_2$ [12]. Thus, we define the visibility sphere radius as the following:

$$r_p = \frac{\lambda_1}{\lambda_2} \cdot d_p = k \cdot d_p. \tag{5}$$

This simple method provides an efficient solution for the determination of visibility sphere radius and avoids the problems resulted from the fixed radius.

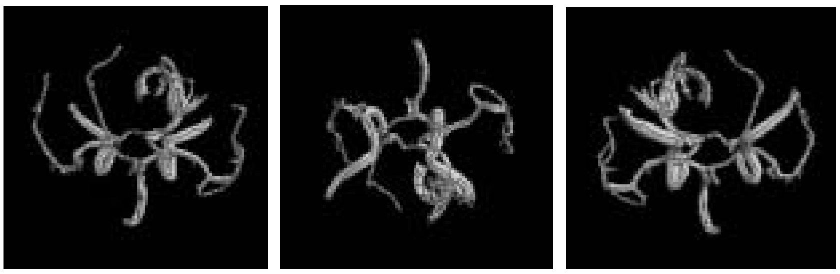


Fig. 5. The shape and centerline of cerebral blood vessel, shown from 3 different angles.

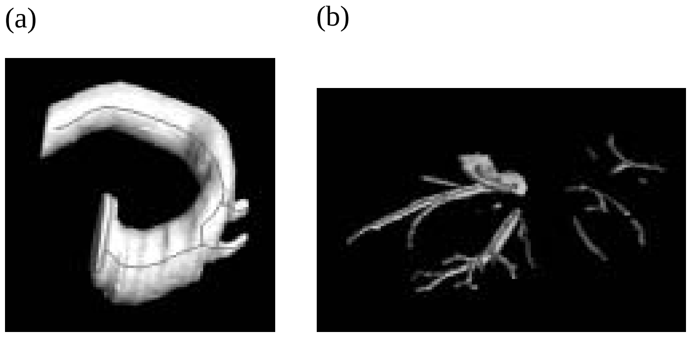


Fig. 6. The shape and centerline of (a) abdominal artery and (b) hepatic blood vessel.

After obtaining all the path points, we use Dijkstra's method to connect them into a path. At the same time, we avoid connecting the points that will penetrate the surface.

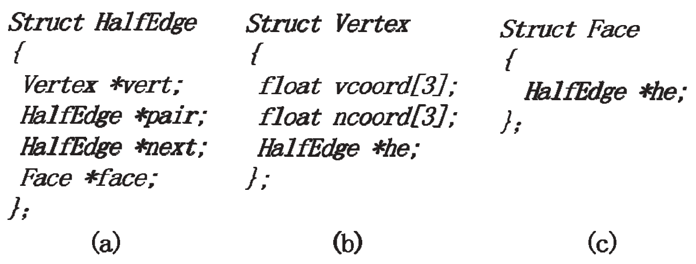
### 3.3. Experimental results

We implement our method with C++ and test it in several different datasets. All of the tests are completed on a PIII 800 MHz PC with 256 MB of main memory and Windows XP. The rendering is left to standard OpenGL API using a GeForce 2 MX-400/32 MB video adapter.

Figure 5 shows the surface model and the centerline of a cerebral blood vessel in three different angles. The data resolution is 256*256*48. There are 6,213 internal voxels and 5,422 boundary voxels. The path consists of 449 voxels. The total run time is 7.32 seconds. Figure 6 shows another two datasets. Figure 6(a) is a fraction of abdominal artery. The data resolution is 512*512*66. There are 238,239 internal voxels and 48,397 boundary voxels. The path consists of 111 points. It costs 57.551 seconds to extract the centerline. Figure 6(b) is a fraction of hepatic blood vessel. The data resolution is 512*512*130. There are 42,062 internal voxels and 20,357 boundary voxels. The path consists of 605 points. The total run time is 106.868 seconds.

## 4. Real-time rendering

We developed a level of detail model based on the view-dependent subdivision, and applied it in virtual endoscopy. For the fraction that the camera sees, we render the detailed model. And for the part that the camera cannot see, we just render the simplified model.



```
Struct HalfEdge        Struct Vertex           Struct Face
{                      {                       {
 Vertex *vert;          float vcoord[3];         HalfEdge *he;
 HalfEdge *pair;        float ncoord[3];       };
 HalfEdge *next;        HalfEdge *he;
 Face *face;           };
};
      (a)                    (b)                     (c)
```

Fig. 7. Structures of half-edge (a), vertex (b), and face (c).

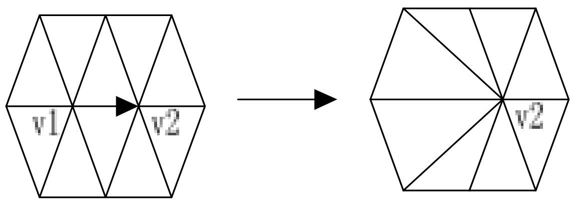


Fig. 8. Half-edge collapse.

Our algorithm consists of two steps. First, the base mesh is obtained by mesh simplification. At the same time, the relationship between the base mesh and the original mesh is recorded through mesh parameterization. Second, navigation was implemented by selective resample when the view parameters were changed. In this step, the adaptive octree was employed so as to make full use of the reusability of the frame-to-frame coherence.

### *4.1. Mesh simplification and parameterization*

Surface simplification is performed to reduce the data amount of the 3-D model. The simplified mesh, which is called base mesh, is to be submitted to the graphics pipeline. As mentioned above, for the part that is out of the view frustum, only base mesh is rendered. For the fraction that is in the view frustum, the base mesh will be subdivided to the original mesh.

We proposed a new mesh simplification algorithm combining half-edge data structure with modified quadric error metric (QEM). Half-edge structure is so flexible that different vertex lists, face lists and half-edge lists can be constructed according to different requirements. In our implementation, we can get all information needed for mesh simplification by adopting several simple structures shown in Fig. 7, and the updates of the adjacency of vertices, faces and half-edges can be easily achieved.

To make full use of the advantages of half-edge structure, we employ half-edge collapse instead of edge collapse, as shown in Fig. 8. The half-edge collapse is similar to edge collapse. The only difference lies in that for half-edge collapse, the start point is pulled into the end point. In other words, the new vertex after the half-edge collapse is the same as the end point of this half-edge. In fact, the half-edge collapse can be viewed as the general edge collapse without locating new vertex. It can also be considered as the vertex removal operator without re-triangulation.

The QEM method proposed by Garland is the most widely used method to simplify meshes [17]. However, it is not suitable for medical surfaces because of the normal vectors. Therefore, we made improvements by integrating the information of normal vectors to the initialization of the simplification process.

Here, we will first briefly introduce the basic principle of QEM. Given a plane that is represented by the standard form

$$n^T v + d = 0 \tag{6}$$

where $n = [a, b, c]^T$ is a unit normal (i.e., $a^2 + b^2 + c^2 = 1$) and $d$ is a scalar constant, the squared distance of a vertex $v = [x, y, z]^T$ from the plane is given by the equation

$$D^2(v) = (n^T v + d)^2 = v^T (nn^T) v + 2(dn)^T v + d^2. \tag{7}$$

Equation (7) is equivalent to the formula

$$D^2(v) = v^T Q v = Q(v), \tag{8}$$

where $v = [x, y, z, 1]^T$ and the quadric Q is treated as a homogeneous matrix

$$Q = \begin{bmatrix} A & b \\ b^T & c \end{bmatrix}, \tag{9}$$

where $A = nn^T$ is a $3 \times 3$ matrix, $b = dn$ is a 3-vector, and $c = d^2$ is a scalar. With this representation, the sum of squared distances to a set of planes can be easily computed:

$$E_Q(v) = \sum_i D_i^2(v) = \sum_i v^T Q_1 v = v^t (\sum_i Q_i) v = v^T Q v \tag{10}$$

where $Q = \sum_i Q_i$. In other words, to compute the squared distances to a set of planes, we only need one quadric that is the sum of the quadrics defined by each of the planes in the set. When contracting the edge $(v_i, v_j)$, the quadric of the new vertex vnew is merely $Q_{\text{new}} = Q_i + Q_j$ and the cost of contraction, which is defined as the sum of squared distances from vnew to the set of planes determined by the adjacent faces of $v_i$ and $v_j$, can be computed through the equation $Q_{\text{new}}(v)_{\text{new}}) = v_{\text{new}} T Q_{\text{new}} v_{\text{new}}$. The point we want to find is the one that minimizes this cost.

The main idea of our improvement is to add the tangential plane of each vertex in the construction of the quadrics. The normal of the tangential plane in vertex $v$ is the unit normal $n$ that is computed through voxel gradient. When constructing the quadrics of $v$, we not only consider the contribution of the adjacent faces of $v$ but also consider the contribution of its tangential plane. The following equation illustrates this idea: $Q_v = \sum_i Q_i + wQ_i$ Where $Q_i$ is the quadric of the $i\text{-}th$ face around vertex $v$, $Q_t$ is the quadric of the tangential plane and $w$ is the weight.

In our approach, the weight $w$ is defined by users. Our tests have proved that if $w$ is too small, it has little influence on simplification procedure while the global results may suffer if $w$ is too large. In our implementation, we select the number of the adjacent faces of vertex $v$ as the weight. Although this approach is very simple, it can indeed improve the visual quality of this kind of 3D medical models as shown in Figs 9(c) and 10(c).

As for mesh parameterization, we use MAPS method. In every simplification step, we computed a 4-tuple to describe the new triangle that the removed vertex is going to be associated with. For the details of MAPS algorithm, refer to [15].

Figures 9 and 10 are two results of our simplification method. In Fig. 9, it took 5 seconds to simplify the abdominal artery model from 142,872 faces to 7,000 faces with our method, while the original QEM method cost 16.414 seconds. In Fig. 10, the knee model was simplified from 95,936 faces to 15,000 faces. Our algorithm cost 4.086 seconds while original QEM cost 12.538 seconds. Furthermore, we can clearly see that our algorithm can get better results than the original QEM method.

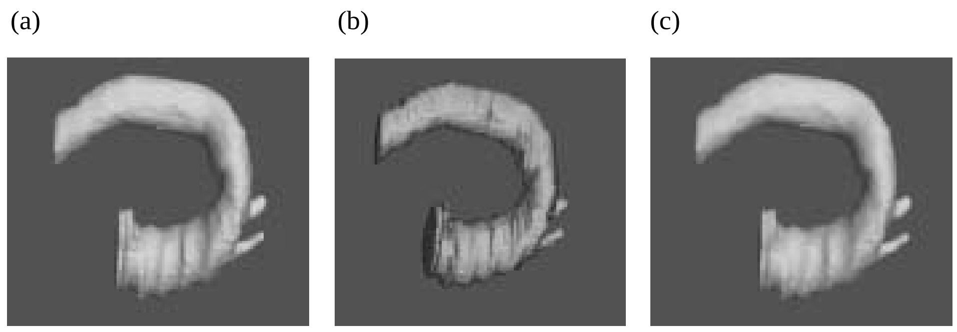


Fig. 9. Simplification of abdominal artery model. (a) Original model (142,872 faces), (b) Simplified model (7,000 faces QEM), and (c) Simplified model (7,000 faces our method).

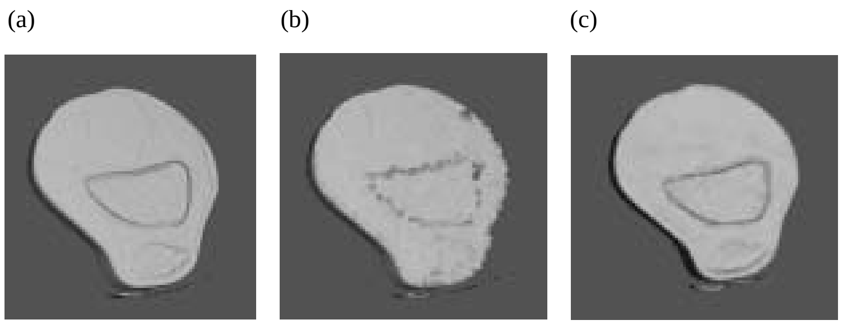


Fig. 10. Simplification of knee model. (a) Original model (95,936 faces), (b) Simplified model (15,000 faces QEM), and (c) Simplified model (15,000 faces our method).

### *4.2. View-dependent Subdivision* (*VDS*)

Hoppe's VDPM traversed each vertex to judge whether it is visible or not, so the speed could not meet the requirement of virtual endoscopy. Based on the fact that the navigation scene of the virtual endoscopy changed slowly, we constructed the index of view parameters with adaptive octree to reduce the search scope. Therefore, we could make full use of the coherence of the adjacent frame, and only deal with the area whose visibility has changed, thus reducing the complexity of the algorithm.

Traditional octree is a uniform spatial partition. However, the distribution of the graphical objects in the boundary box is not uniform, which will result in the wastage of the storage because many leave nodes are empty. So we used an adaptive octree data structure to record the triangles. We defined the maximum number of the triangles in each leaf node. If the triangle number of one node exceeded this value, we partitioned it until all the triangle number in each node was lower than the value. Thus, the adaptive octree data structure was built.

After the adaptive octree had been built, we used it to judge whether the object is in the view frustum or not. First, we traversed the octree from the root node. If the area associated with the root node was all in the view frustum or all outside of the view frustum, the entire model was visible or invisible. Otherwise, we traversed the octree in depth, and judged the visibility of each node until reaching the leaf node. If the leaf node was in the view frustum, then all the triangles of this node were considered to be in the frustum. If it also meets the criterion of surface orientation and screen-space geometric error, then we will subdivide these triangles. Otherwise, all the triangles of the leaf node are invisible and we performed down sampling.

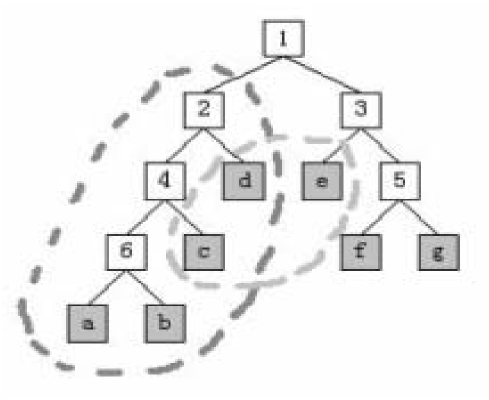


Fig. 11. The partition of the leaf nodes.

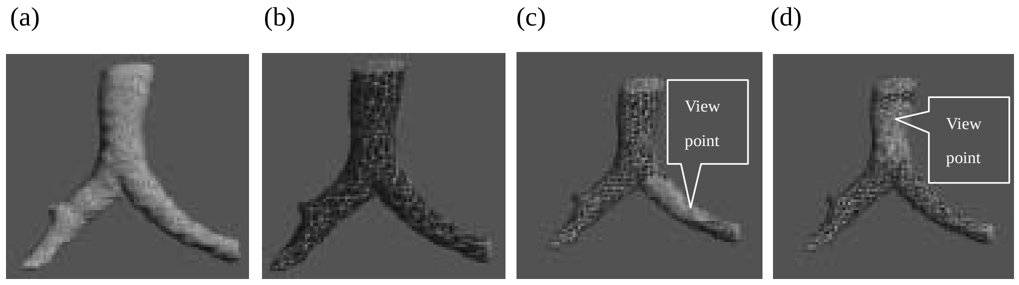


Fig. 12. View-dependent subdivision for trachea model. (a) Original model (26,979 faces), (b) Simplified model ((1,500 faces), (c) View-dependent Model (2,313 faces, and (d) View-dependent Model (6,397 faces).

Because the navigation scene of virtual endoscopy changed slowly, the view parameter did not change quickly. So a big cross area was observed between the adjacent view frustums. We considered only the area which had been changed. In the octree, each specific view parameter divided all the leaf nodes into two sets: visible and invisible. Each change of the view parameter led to the repartition of the leaf nodes. Figure 11 shows the example of the repartition of the nodes (For convenient, we use the binary tree, and the octree can be extended similarly). The dashed lines outlined the visible sets of the adjacent view frustum, $\{c, d, e\}$ and $\{a, b, c, d\}$. In the case that the view frustum changed from the light color area to the dark color area, we only deleted the element $e$ by performing down sampling and restored the node $a$ and $b$ by performing subdivision. $c$, $d$, $f$ and $g$ did not change with the view parameter, so we did not deal with them.

Figures 11 and 12 are two examples of our view-dependent rendering method for trachea model and simulated colon model.

## 4.3. Virtual navigation

With the centerline and fairing surface, the navigation can be easily implemented using our rendering techniques. For an efficient interactive navigation, two properties must be satisfied. One is that the physician can intuitively and easily change the camera position and direction. The other is that the camera should never penetrate through the surface, even when incorrectly handled by the physician [5]. The former property can be easily satisfied with OpenGL. As for the latter, many techniques have

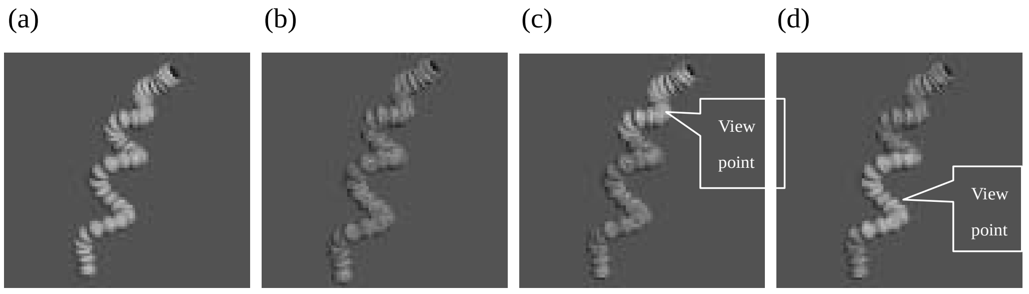


Fig. 13. View-dependent subdivision for simulated colon model. (a) Original model (30,720 faces), (b) Simplified model (8,000 faces), (c) View-dependent Model (16,862 faces), and (d) View-dependent Model (17,315 faces).

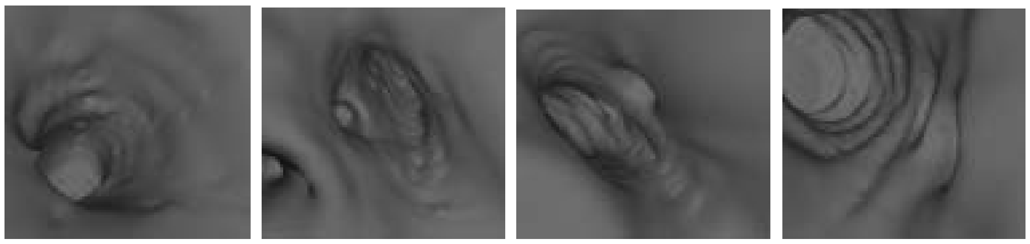


Fig. 14. Four navigation frames inside the abdominal artery.

been proposed such as collision detection and physical-based model [5]. In this paper, we use a very simple method to solve the problem. As we know, image segmentation classified the volume into three categories: external voxels, internal voxels and boundary voxels. We set a flag to each voxel indicating which category the voxel belongs to. When the camera moves during navigation, we can justify whether the current camera position belongs to internal voxel or not. Therefore, the penetrating through the surface can be avoided. In Fig. 14, some navigation frames inside the abdominal artery are given. Figure 15 shows a whole process of interactive virtual navigation. Every frame in Fig. 15 consists of two parts. The left is the scene inside the model, and the right shows the camera position and the view direction.

## 5. Conclusion

In this paper, we presented a generic scheme for virtual endoscopy which consists of fairing surface generation, flight-path planning and advanced real time rendering techniques. First, the normal vectors of reconstructed model through Marching Cubes were smoothed through Laplacian method. Then the centerline was extracted with our new path planning algorithm. We combined Hessian Matrix with distance transformation for automatic centerline extraction, using the eigenvalues of Hessian Matrix to adaptively determine the visibility sphere radius. Furthermore, we developed a new scheme for real time rendering, which consists of modified mesh simplification algorithm, mesh parameterization and view-dependent subdivision. During navigation, we used the dynamic model, and only dealt with the area in which the visibility had been changed. Therefore, complexity of the algorithm is reduced.

In future work, we will focus on navigation techniques. One direction is to integrate sqrt(3) subdivision [18] or 4–8 subdivision [19] to our framework because in butterfly subdivision that we are now using, the number of triangles will increase four times in each subdivision, which makes rendering

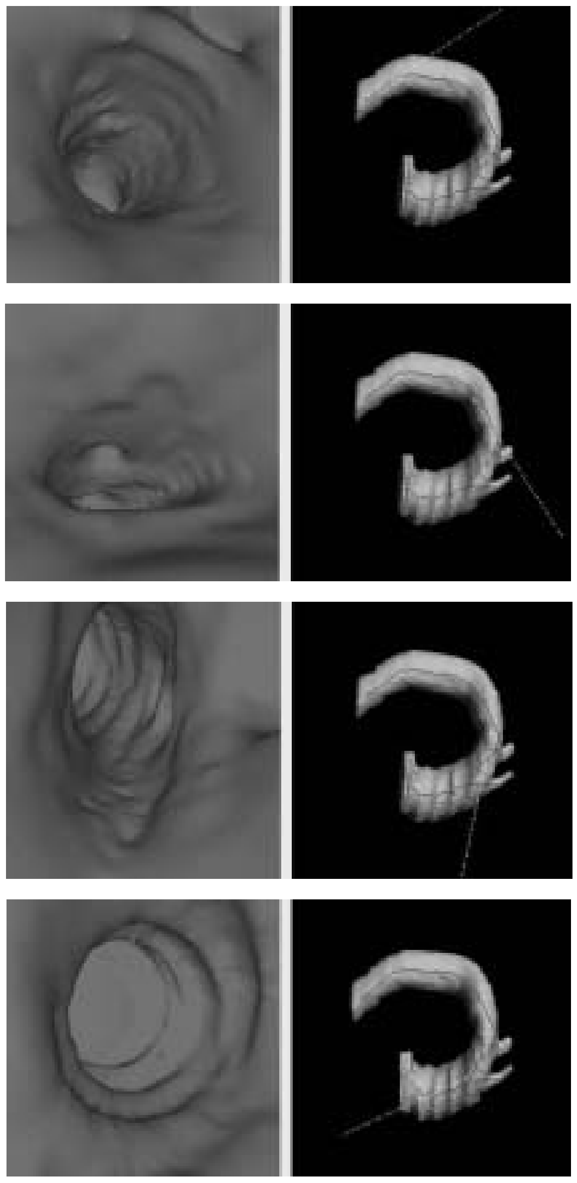


Fig. 15. A whole process of interactive virtual navigation.

difficult. Furthermore, we can also add cubic mapping to our system to make users observe the surface more clearly in a panoramic way.

## Acknowledgement

This paper is supported by the Project for National Science Fund for Distinguished Young Scholars of China under Grant No. 60225008, the Special Project of National Grand Fundamental Research 973 Program of China under Grant No. 2002CCA03900, the National High Technology Development

Program of China under Grant No. 2002AA234051, and by the National Natural Science Foundation of China under Grant Nos. 30370418, 90209008, 60302016, 60172057, 30270403.

## References

[1] M. Wan, Z. Liang, Q. Ke, L. Hong, I. Bitter and A. Kaufman, Automatic Centerline Extraction for Virtual Colonoscopy, *IEEE Transactions on Medical Imaging* **21**(12) (2002), 1450–1460.

[2] M. Zhao, J. Tian, H. He and G. Li, A Surface Reconstruction Algorithm for High Resolution Medical Images, Proceedings of International Conferences on CAD&CG2001, *International Academic Publishers* **1** (2001), 286–290, Kunming, China.

[3] S. Gibson, *Using distance maps for accurate surface representation in sampled volumes*, In Proceedings of the 1998 Symposium on Volume Visualization, ACM SIGGRAPH, October 1998, pp. 23–30.

[4] D.A. Field, Laplacian smoothing and Delaunay triangulations, *Commuications in Applied Numerical Methods* **4** (1998), 709–712.

[5] L. Hong, S. Muraki, A. Kaufman, D. Bartz and T. He, Virtual Voyage: Interactive Navigation in the Human Colon, in: *Proceedings of SIGGRAPH'97*, T. Whitted. Los Angeles: ACM Press, 1997, pp. 27–34.

[6] T. Pavlidis, A Thinning Algorithm for Discrete Binary Images, *Computer Graphics and Image Processing* **13**(2) (1980), 142–157.

[7] T. He, L. Hong, D. Chen and Z. Liang, Reliable Path for Virtual Endoscopy: Ensuring Complete Examination of Human Organs, *IEEE Transactions on Visualization and Computer Graphics* **7**(4) (2001), 333–342.

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

[9] M. Wan, F. Dachille and A. Kaufman, Distance-Field Based Skeletons for Virtual Navigation, in: *Proceedings of IEEE Visualization'01*, San Diego, CA: 2001, pp. 239–246.

[10] T. Saito and J. Toriwaki, New Algorithms for Euclidean Distance Transformation of an N-Dimensional Digitized Picture with Applications, *Pattern Recognition* **27**(11) (1994), 1551–1565.

[11] J. Hladuvka, A. König and E. Gröller, Exploiting Eigenvalues of the Hessian Matrix for Volume Decimation, in: *conference proceedings of the 9th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2001, WSCG'2001, University of West Bohemia, Pilsen, Czech Republic*, (Vol. 1), S. Vaclav, ed., February 2001, pp. 124–129.

[12] S.R. Aylward and B. Elizabeth, Initialization, Noise, Singularities, and Scale in Height Ridge Traversal for Tubular Object Centerline Extraction, *IEEE Transactions on Medical Imaging* **21**(2) (2002), 61–75.

[13] H. Hugues, *Progressive Meshes*, Proc. of SIGGRAPH'96, pp. 99–108.

[14] H. Hugues, *View-Dependent Refinement of Progressive Meshes*, Computer Graphics Proceedings, Annual Conference Series, SIGGRAPH, Los Angeles, California, 1997, pp. 189–198.

[15] A.W.F. Lee, W. Sweldens, P. Schroder, L. Cowsar and D. Dobkin, MAPS: Multiresolution adaptive parameterization of surfaces, Computer Graphics Proceedings, *SIGGRAPH* **98** (1998), 95–104.

[16] H. He, J. Tian, M. Zhao and G. Li, New continuous level-of-detail algorithm and its application in virtual endoscopy. Oral Report, in: *Proceedings of SPIE Medical Imaging 2003,* San Diego, USA, February, 2003.

[17] C. Michael and P.S. Heckbert, *Surface simplification using quadric error metric*, Proc. of SIGGRAPH'97, 209–216.

[18] L. Kobbelt, *Sqrt*(*3*) *Subdivision*, In Computer Graphics Proceedings, ACM Siggraph 2000, pp. 103–112.

[19] L. Velho and D. Zorin, *4–8 Subdivision*, CAGD, Special Issue on Subdivision Techniques 2001.