JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 33, 1213-1235 (2017)
DOI: 10.6688/JISE.2017.33.5.7

Semantic Segmentation and Structure Representation of the Generalized Body Cavity

JIAN-QING MO, HAN-WU HE+, JIN-FANG LI AND YU-WEI WEI
Guangdong Provincial Key Laboratory of CIMS
Guangdong University of Technology
Guangzhou, 510006 P.R. China
E-mail: {momolon; hwhe; lijinfang; weiyuwei}@gdut.edu.cn

Manual pre-operative planning is time consuming and depends mainly on the clinical experience of surgeons. It is helpful for automatic pre-operative planning and robotic surgery to explore the systematic methodology for operative environment analysis and understanding. Since shape segmentation, structure extraction and representation are fundamental to shape understanding, this paper aims to explore the semantic segmentation algorithm and the structure representation of the generalized body cavity. Our general strategy for abstract shape segmentation is the explicit boundary method. The cutting contours combining critical loops and local feature contours are constructed to slice the input mesh. To achieve reasonable critical loops, we first defined a real function that is invariant to translation, rotation and scaling transformation. Then, an efficient segmentation algorithm was proposed to obtain approving semantic segmentation. In addition, we presented a structure representation that was expected to be applied to the path planning of surgery. Finally, the articular cavity of human knee joint was taken as an example to verify the proposed approach.

Keywords: pre-operative planning, shape segmentation, structure representation, sub-cavity network, cutting contour

1. INTRODUCTION

Pre-operative planning plays an important role in improving surgical accuracy, shortening operation time and improving surgical outcomes. Therefore, much effort has been made to seek feasible approaches for pre-operative planning. Thanks to the significant advances in the techniques of medical imaging and three-dimensional visualization, an increasing number of planning systems have been developed. Although those systems can often provide detailed insight into anatomical structures, and support quantitative analysis to a certain extent, excessive dependency on the clinical experience and the manual work of surgeons has not been greatly improved yet. Recently, automatic pre-operative planning [1-3] has received attention, and further investigation has been performed to alleviate the manual effort of surgeons, or to adapt for computer-assisted robotic surgery. However, previous studies focused mainly on specific applications, and rare systematic research on automatic pre-operative planning approach has been reported. Therefore, we are currently engaged in a project designed to set up a systematic methodology for automatic pre-operative planning that is based on operative environment analysis and understanding. Fig. 1 describes the framework of the system, and this work focus on the topology analysis.

Received December 21, 2015; revised April 7 & September 19, 2016; accepted December 10, 2016.
Communicated by Yung-Yu Chuang.
+ Corresponding author.

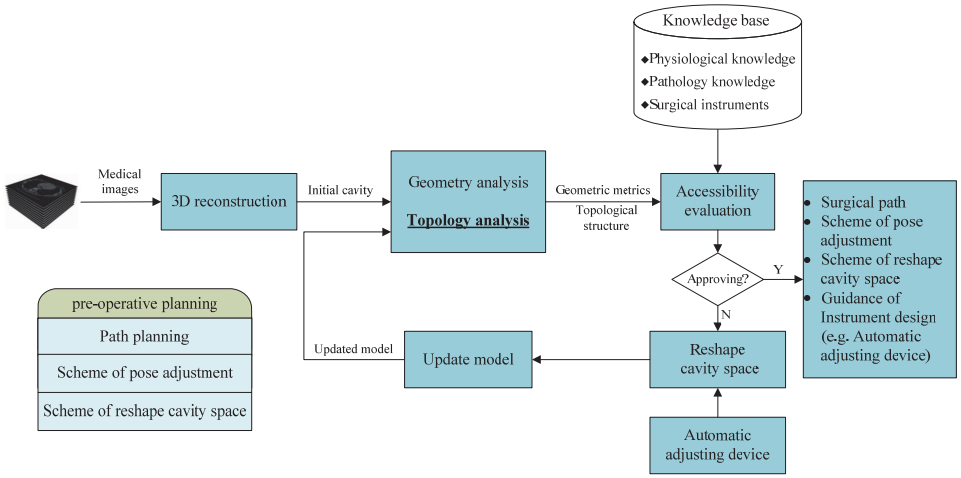


Fig. 1. The framework of the pre-operative planning system.

The traditional path planning algorithms are often inefficient because they require intersection testing with the scattered organs. In addition, they are incapable of exploring the characteristics of cavity space. Different from the existing works, the body cavity is the very environment to be analyzed quantitatively in our work. The body cavity in our work refers not only to the natural body cavity but also to any anatomic structure that served as a candidate inside which an artificial channel is to be created. As such, we named it *generalized body cavity*. The body cavity is enclosed by soft tissues and/or bony tissues, and lesions distribute inside or around these tissues. The quantitative analysis of the body cavity is important and viable for automatic pre-operative planning for two reasons. First, the cavity space is the movement space of surgical instruments; and second, it encodes much information of its surrounding tissues, such as geometrical and topological information. However, the quantitative analysis of body cavities is a significantly challenging issue because they are often irregular, complex, and even separated into discontinuous spaces by constraint tissues.

Since shape segmentation is fundamental to shape analysis and understanding, we focus our attention on the method of shape segmentation in which specific application context is taken into account. According to [4], there are two principal types of shape segmentation applied in different application contexts. One is part-type segmentation, and the other is surface-type segmentation. Our application falls into the former one since our goal is to decompose the body cavity into different parts. However, we noticed that there were some differences between the existing methods of part-type segmentation and our method due to the different goals. The former was dedicated to subdivide shapes into 'functional parts' to meet the requirement of reverse engineering, animation or 3D retrieval, *e.g.*, to subdivide an animal into head, torso and limbs. Our goal is to decompose the body cavity into different 'semantic parts' because there are no natural function parts in the body cavity. The 'semantic' in this paper is associated with the abstract constraint conditions that are determined by the specific context of segmentation.

The previous algorithms for part-type shape segmentation depend mainly on the human perception, so they were designed to subdivide those objects either with clear

hierarchy or with obvious contour between their functional components. However, for those abstract shapes without these features, it is hard to achieve the approving segmentation by applying these algorithms. To address this issue, we presented an explicit boundary method for abstract shape segmentation in which the semantic constraint is the boundary of branches. The recognition and extraction of boundaries rely on geometrical information and topological information as well. We also present a graph-like structure – sub-cavity network (SCN) – to encode geometrical, topological and physiological information. It is expected to be applied to automatic surgical path planning, robotic surgery and to guide the design of surgical instruments. Our approach was evaluated in knee articulation.

2. RELATED WORK

In the past two decades, much research has been carried out on shape segmentation for a single object [4]. More recently, methods for segmenting a set of objects has received more attention. However, since the data of an individualized patient are used to shape analysis in our application, we still focus on the method that can be utilized to subdivide an individual object into semantic parts.

A variety of algorithms for the part-type shape segmentation of single objects have been proposed. Various classifications exist according to different criteria, such as the *geometric-based method* and the *topology-based method*, the *region-based method* and the *explicit boundary method*, the *surface-based method* and the *volume-based method*.

Although the region-based method can be applied to part-type segmentation, especially when applying the volumetric information as the SDF (the shape diameter function) in [12], previous algorithms for part-type segmentation rely mainly on the explicit boundary method [13-16]. The explicit boundary method first constructs boundary contours in some way, for instance, extracting feature contours [13], sampling isolines [15], or using a statistical-based method [14]. The main concern of this stage is the credibility of component boundaries, which aims to guarantee the segmented parts are natural and consistent with the human visual perception. Prevalent approaches are built upon the *minima rule* which bridges the human perception and the surface properties of shapes, for instance, the lower-level surface metric curvature [13], or volumetric metrics VSI [17] and MSP [18], and the high-level properties convexity [19] or concavity [15]. A quantitative evaluation of these algorithms was provided by the Princeton benchmark [20]. It has proven that the algorithms with high-level properties, or with multiple optimization in a set of cutting boundary candidates, such as the randomized cuts method [14] and the concavity-aware method [15], could yield better segmentation results. By contrast, the concavity-aware method is more efficient and more easily implemented as well, since it generates cutting boundary candidates by sampling isolines from the concavity-aware fields rather than generating a random set of segmentations via several existing segmentation methods.

The explicit boundary method usually requires high-quality input mesh, such as watertight and manifold; special processing is often needed to ensure the robustness of the algorithm, and it cannot handle incomplete shapes. Moreover, the topological properties that are important in our application were seldom considered in the previous algorithms for shape segmentation.

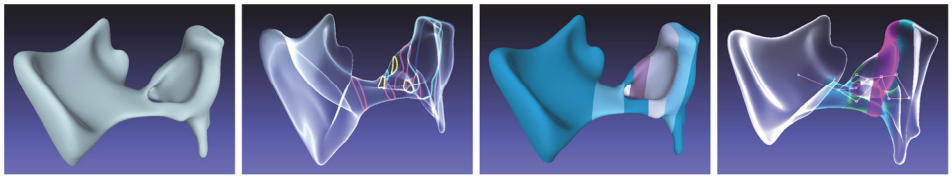
Within the last decade, an approach has emerged for shape segmentation based on the Morse theory, which combines the topological and geometrical properties of an input surface via the real functions defined on shapes [21]. In [22], to achieve object decomposition, the Reeb graph rooted in Morse theory was used to extract the structural and topological information of a mesh surface. However, similar to the common sweeping techniques, the approach for the construction the Reeb graph in [22, 23] is time-consuming. Noticing that the topological changes of level sets occur only at critical points, Giuseppe Patane *et al.* [24] proposed an efficient contouring algorithm for constructing the Reeb graph without sampling, sweeping, or sorting the function values.

Although the part-type shape segmentation has made great progress, it is still a challenging problem. The previous algorithms are more suitable to subdivide those objects with clear contours between their parts or with natural functional parts. Moreover, accuracy, efficiency and robustness cannot be satisfied by using only one of these algorithms.

Structural information is also helpful for shape understanding. The graph-like structures, such as the skeleton, contour tree and Reeb graph, are important structure representations. They can encode the geometrical or topological information of a 3D shape in an intuitive manner and represent the connected relationships between shape parts as well. The semantic annotation [25] can enrich the information of each sub-part and expand their potential applications.

3. OVERVIEW

We will present a graph-like structure representation (SCN) of a given shape. The construction of an SCN is based on the result of shape segmentation using the explicit boundary method. Fig. 2 describes the procedure of constructing the SCN.



(a) (b) (c) (d)

Fig. 2. The procedure of constructing an SCN; (a) The original mesh model; (b) Construction of cutting contours; (c) Subdividing the shape via cutting contours; (d) Construction of the SCN.

The SCN is expected to be applied to automatic surgical path planning and robotic surgery, as well as to guide the design of surgical instruments, which is based on three reasons: (1) as shown in Figs. 7 and 12, the SCN provides all candidate paths from the source to the target; (2) the information encoded in the nodes and arcs of the SCN can be used in the accessibility evaluation, which is the basis for generating the optimal path; (3) the shape and the topology structure of one cavity may vary under different conditions. We use the SCN to study the laws of these varieties, and then design the adjusting de-

vices according to the laws. As we will describe in section 5, the pose adjusting device can be used to assist autonomous robotic surgery.

The proposed approach consists of the following steps.

Definition of real function *Cutting contours* used to subdivide a shape into parts may consist of *critical loops* and *feature lines*. The choice of real function is a key factor, as it can determine the number and location of critical points, and determine the points through which a critical loop passes. In our specific application, we defined a real function that was based on the principal direction projection (section 4.1).

Construction of cutting contours For a given meshed manifold M and the real function f, we compute the critical points of f with the method in [26], and create critical loops with the approach in [24]. Although critical loops are controllable, they often deviate from the feature contours. So we construct feature lines and combine them with critical loops according to certain rules when necessary.

Shell traversal *Shell traversal* is a process of assigning triangles to certain groups with respect to the cutting contours. A shell (or part) is a set of triangles that consist of the original triangles of mesh M and the boundary fragments generated by triangle splitting as shown in Fig. 4. We proposed an efficient strategy – *dynamic authorities granted strategy* (DAGS, see section 4.3) for shell traversal.

Construction of a sub-cavity network As part of the whole cavity, a *sub-cavity* is obtained by mesh segmentation. In section 4.4, we defined a graph-like structure – *sub-cavity network*. It reflects the connectivity relationship among sub-cavities and encodes geometrical, topological and physiological information. Then, we will present an algorithm that requires only the adjacency information of shells for constructing SCN.

4. METHODS

4.1 Definition of Real Function

The key to our application lies in extracting the topological information. It reveals how the parts of the cavity are separated by the constraint tissues. This information is expected to be used to guide surgical instruments to bypass the constraint tissues before reaching the pre-specified region. Therefore, our definition of real function is associated with the movement direction of surgical instruments. A direction (*e.g.*, the dominant movement direction of surgical instruments) is chosen as the height direction and then the heights of all the vertices of M are computed with respect to this direction. That is, by mapping the original coordinates of vertex v_i (x_i, y_i, z_i) to a new coordinate system, we will obtain the height value that is regarded as the function value of vertex v_i. Then, the real function f defined on a given mesh M can be expressed as

$$f(v_i) := h(T(x_i, y_i, z_i)), \tag{1}$$

where T denotes the translation of coordinates.

The values of function f are invariant under translation or rotation transformation, and they are also invariant under scaling transformation by normalizing the function values.

Projection transformation Fig. 3 depicts the new coordinate system $ox_sy_sz_s$ which depends on the three orthogonal unit vectors:

$$\begin{cases} \hat{\boldsymbol{\alpha}} := [x_\alpha, y_\alpha, z_\alpha]^T \\ \hat{\boldsymbol{h}} := [x_h, y_h, z_h]^T \\ \hat{\boldsymbol{v}} := [x_v, y_v, z_v]^T \end{cases}. \tag{2}$$

They represent the z_s-axis (the height direction), x_s-axis and y_s-axis, respectively. Vector $\hat{\boldsymbol{\alpha}}$ is given in advance or pre-computed automatically. However, $\hat{\boldsymbol{h}}$ and $\hat{\boldsymbol{v}}$ need to be evaluated by the given vector $\hat{\boldsymbol{\alpha}}$ according to the Eq. $\hat{\boldsymbol{\alpha}} = \hat{\boldsymbol{h}} \times \hat{\boldsymbol{v}}$.

When $x_\alpha = 0$, we set

$$\begin{cases} \hat{\boldsymbol{h}} = [1, 0, 0]^T \\ \hat{\boldsymbol{v}} = \left[0, z_\alpha \Big/ \sqrt{y_\alpha^2 + z_\alpha^2}, -y_\alpha \Big/ \sqrt{y_\alpha^2 + z_\alpha^2}\right]^T \end{cases}. \tag{3}$$

Otherwise, set

$$\begin{cases} \hat{\boldsymbol{h}} = \left[-z_\alpha / \|\boldsymbol{h}\|, 0, x_\alpha / \|\boldsymbol{h}\|\right]^T \\ \hat{\boldsymbol{v}} = \left[x_\alpha y_\alpha / \|\boldsymbol{v}\|, -\left(x_\alpha^2 + z_\alpha^2\right) / \|\boldsymbol{v}\|, y_\alpha z_\alpha / \|\boldsymbol{v}\|\right]^T \end{cases}, \tag{4}$$

where

$$\begin{cases} \|\boldsymbol{h}\| = \sqrt{x_\alpha^2 + z_\alpha^2} \\ \|\boldsymbol{v}\| = \sqrt{\left(x_\alpha^2 + z_\alpha^2\right)\left(x_\alpha^2 + y_\alpha^2 + z_\alpha^2\right)} \end{cases}. \tag{5}$$

As shown in Fig. 3, P_i (x_i, y_i, z_i) is a vertex of mesh M. According to the fact that the sum of a closed chain of vectors is zero, we have

$$-\overrightarrow{oP_i} + x_{is}\hat{\boldsymbol{h}} + y_{is}\hat{\boldsymbol{v}} + z_{is}\hat{\boldsymbol{\alpha}} = 0. \tag{6}$$

Then, the new coordinates (x_{is},y_{is},z_{is}) of P_i in $ox_sy_sz_s$ are expressed by

$$\begin{bmatrix} x_{is} \\ y_{is} \\ z_{is} \end{bmatrix} = \begin{bmatrix} x_h & x_v & x_\alpha \\ y_h & y_v & y_\alpha \\ z_h & z_v & z_\alpha \end{bmatrix}^{-1} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}. \tag{7}$$

Directional compactness and global principal direction Although the projection

direction $\hat{\boldsymbol{\alpha}}$ can be specified by the user, for easy implementation, it is necessary to find an approach to automatically compute a reasonable height direction. Our approach to solve this problem will be represented after introducing the definition of *directional compactness*.

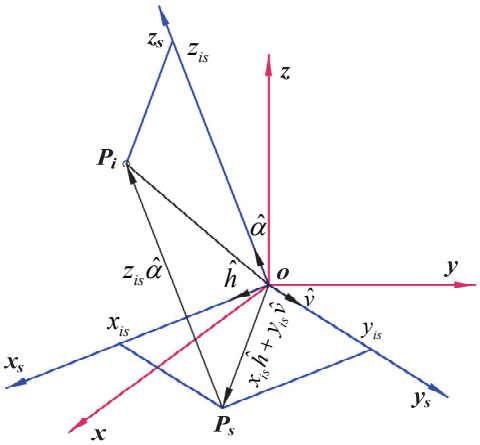


Fig. 3. The original coordinate system $oxyz$ and the new coordinate system $ox_sy_sz_s$.

Project all the vertexes of M onto a plane (*e.g.*, the coordinate plane x_soy_s in Fig. 3) that is perpendicular to the given vector $\hat{\boldsymbol{\alpha}}$. The compactness degree of all the projections is called directional compactness (DC) with respect to the vector $\hat{\boldsymbol{\alpha}}$. The quantitative calculation of DC is defined as

$$DC := \left(\sum_{i=1}^{n} d_{iE} \right)^{-1}, \tag{8}$$

where n is the number of vertexes on M; and d_{iE} represents the distance from the projective point $P_s(x_{is}, y_{is})$ to the projective center $E(\bar{x}_s, \bar{y}_s)$. d_{iE}, $\bar{x}_s$ and $\bar{y}_s$ are defined as

$$d_{iE} := \sqrt{(x_{is} - \bar{x}_s)^2 + (y_{is} - \bar{y}_s)^2}, \tag{9}$$

$$\bar{x}_s = \frac{1}{n}\sum_{i=1}^{n} x_{is}, \ \bar{y}_s = \frac{1}{n}\sum_{i=1}^{n} y_{is}. \tag{10}$$

DC reflects the distribution of cavity space along a certain direction. A large DC indicates that surgical tools can access much cavity space and surrounding lesions even if their poses do not adjust too much. Consequently, we define the projective direction in which a maximum DC is yielded as *global principal direction* (GPD). Thus, the DC and the GPD have potential value for guiding the planning of a surgical path.

Computation of GPD It is an optimal problem to determine the GPD:

$$\begin{aligned} &Max(DC(\hat{\boldsymbol{\alpha}})) \\ &s.t. \quad \| \hat{\boldsymbol{\alpha}} \| = 1, and \quad y_\alpha \geq 0 \end{aligned} \tag{11}$$

The value space of the unit vector $\hat{\boldsymbol{\alpha}}(x_\alpha, y_\alpha, z_\alpha)$ is a unit sphere. According to the central symmetry, the value space is limited in the scope of hemisphere. A global search algorithm can be used to calculate $\hat{\boldsymbol{\alpha}}$, but it is less efficient. An alternative method is to introduce the *WPCA* (*weighted principal components analysis*) method [27].

Select the coordinate plane $x_s o y_s$ shown in Fig. 3 as the projective plane; according to the Eq. (7), we have

$$\begin{bmatrix} x_{1s} & \cdots & x_{ns} \\ y_{1s} & \cdots & y_{ns} \\ z_{1s} & \cdots & z_{ns} \end{bmatrix} = \begin{bmatrix} x_h & x_v & x_\alpha \\ y_h & y_v & y_\alpha \\ z_h & z_v & z_\alpha \end{bmatrix}^{-1} \begin{bmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \\ z_1 & \cdots & z_n \end{bmatrix}. \tag{12}$$

Let

$$\boldsymbol{P}_S = \begin{bmatrix} x_{1s} & \cdots & x_{ns} \\ y_{1s} & \cdots & y_{ns} \\ z_{1s} & \cdots & z_{ns} \end{bmatrix}, \quad \boldsymbol{T} = \begin{bmatrix} x_h & x_v & x_\alpha \\ y_h & y_v & y_\alpha \\ z_h & z_v & z_\alpha \end{bmatrix}^{-1}, \quad \boldsymbol{P} = \begin{bmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \\ z_1 & \cdots & z_n \end{bmatrix}. \tag{13}$$

Then, we have

$$\boldsymbol{P}_S = \boldsymbol{T}\boldsymbol{P}, \tag{14}$$

where $\boldsymbol{P}_S$ is a linear combination of the original coordinates of the n vertices, and $\boldsymbol{T}$ is a linear transformation between $\boldsymbol{P}$ and $\boldsymbol{P}_S$.

According to the definition of DC, a maximum DC is needed to make the projective points as concentrated as possible by looking for a projective transformation $\boldsymbol{T}$. This means that the coordinates of the projections have a smaller variation range on X_s and Y_s than on Z_s. Therefore, the matrix $\boldsymbol{T}$ can be determined by applying the WPCA method. The approach is efficient because it does not require calculating DC directly, but analyzes only the original coordinates. For the same mesh, our experiments showed that the result of the WPCA method is almost identical to that of the global search algorithm.

4.2 Construction of Cutting Contours

In this section, we will discuss the approach of constructing cutting lines which may consist of critical loops and feature lines.

Computing critical loops Next we will briefly review the method of computing critical loops; for more details, we refer readers to [24]. Suppose that f: $\boldsymbol{M} \to \mathbb{R}$ is a piecewise-linear real function that is defined on the vertices of a closed 2-manifold triangle mesh M. The critical loop is the isocontour passing through a saddle point V_s with respect to the function values, that is, for any point p_i that belongs to the critical loop L_i at saddle point V_s, there exists that $f(p_i)$ equals to $f(V_s)$. Therefore, starting from the one-star (Fig. 4) of V_s, iteratively solve the intersection points of critical loop L_i and the edges

of the triangle on the mesh by linear interpolation (Fig. 4). The iteration proceeds by using the triangle-triangle adjacencies until L_i is closed at V_s. The intersection point p_{ij} between the critical loop L_i and the edge (v_i, v_j) is computed by

$$p_{ij} := (1-t)v_i + tv_j, \tag{15}$$

where t ($0 \le t \le 1$) is a real number defined as

$$t := \frac{f(p_{ij}) - f(v_i)}{f(v_j) - f(v_i)} = \frac{f(V_s) - f(v_i)}{f(v_j) - f(v_i)}. \tag{16}$$

Fig. 4 illustrates the computation method of critical loop. The triangle $v_1v_2v_3$ is split into three boundary triangles (we call them *boundary fragments* because they are borders of two adjacent parts) by the critical loop crossing over it. We classified them into two categories – *lower boundary fragment* (LBF) and *upper boundary fragment* (UBF) – according to relative function value. If the function value of at least one vertex in a fragment is less than that of the corresponding saddle, then it is a lower boundary fragment; on the contrary, if the function value of at least one vertex in a fragment is greater than that of the corresponding saddle, it is an upper boundary fragment.

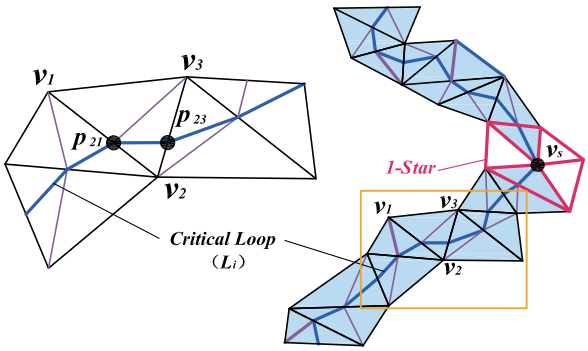


Fig. 4. Computing a critical loop. The triangle $v_1v_2v_3$ is cut into three *boundary fragments*, $v_1p_{21}v_3$, $v_2p_{23}p_{21}$ and $v_3p_{21}p_{23}$, by the critical loop. If $v_2p_{23}p_{21}$ is a *lower boundary fragment* (*LBF*), the other two are *upper boundary fragments* (*UBF*).

Constructing feature contours The critical loops have good controllability because they are perpendicular to the GPD. However, they are usually not coincident with the feature contours, which could make the subsequent accessibility evaluation more complicated. The accessibility evaluation of the sub-cavity requires prior knowledge of its adjacent tissues, which are characterized as feature lines. We merge the feature lines l_i into the cutting contour c_i, which means the sub-cavities obtained by c_i are bounded by l_i. It can limit the numbers of sub-cavities related to a certain tissue to simplify the accessibility evaluation. So we construct the feature lines and then combine them with the critical loops according to certain rules when necessary.

Our method includes the following steps: (1) extract feature edges; (2) filter feature edges; (3) traverse feature edges from a selected seed; (4) prune; (5) close feature contours and (6) eliminate local rings.

Similar to [28], we use the dihedral angles as feature values to detect feature edges. Limited by the paper's length, we omit the details of constructing the feature contours.

Combining critical loop and feature contour The conditions of combination are given as follows.

(1) One critical loop intersects with one feature contour at one or more points, and part of them is similar; or
(2) One critical loop does not intersect with any feature contour, but the distance between a part of one critical loop and one feature contour is less than a given threshold.

If one of the above two conditions is satisfied, some parts of critical loops will be replaced by some parts of the corresponding feature contours.

For the critical loop $L = \bigcup_{i=1,m} L_i$ and the feature line $l = \bigcup_{i=1,n} l_i$, if they meet the first condition, suppose $L_i(1 \le i \le m)$ is similar to $l_j(1 \le j \le n)$, then replace L_i with l_j, and we obtain the cutting contour $c_i = (L - L_i) \cup l_j$; otherwise, if they meet the second condition, suppose the distance between $L_i(1 \le i \le m)$ and $l_j(1 \le j \le n$ is less than a given threshold, then replace L_i with l_j, and connect their corresponding endpoints, and we obtain the cutting contour $c_i = (L - L_i) \cup l_j \cup \{p_0, v_0\} \cup \{p_1, v_1\}$, where the endpoint set is $\{p_0, p_1\} \subset L - L_i$, $\{v_0, v_1\} \subset l_j$.

4.3 Shells Traversal Strategy

The cutting contours serve as the barriers of shells. They are initially granted authorities both to upward traversal and to downward traversal. Additionally, the authorities may be updated dynamically during the shells traversal. Therefore, we named this approach *dynamic authorities granted strategy*. The algorithms of shells traversal and one pass traversal are shown in Algorithms 1 and 2, respectively.

We defined two types of triangles first, namely, a c*rossing triangle* and a *non-crossing triangle*. A crossing triangle is split by one or more cutting contours, but no cutting contour crosses over a non-crossing triangle.

Algorithm 1: Shells traversal applied DAGS.

```
1: for each cutting contour C_i do
2:     D ← getAllowedDirection(C_i)
3:     if verify(D)=true then
4:         S# ← number of currentShell S# +1
5:         T_i ← getInitialTriangle(D, C_i)
6:         traverseCurShell(D, C_i, T_i)
7:         disableDirection (D, C_i)
8:     end if
9: end for
```

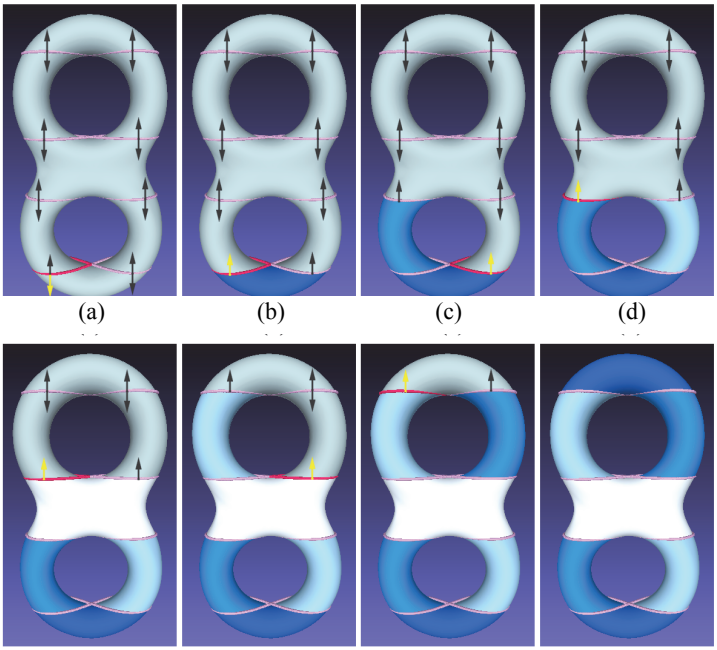


Fig. 5. From (a) to (h) illustrates the shells traversal by DAGS. The highlight contour is selected as the starting in one pass traversal, and the arrows mean the allowed traversal direction which may be updated during traversal.

Fig. 5 illustrates the shells traversal of a bi-torus by DAGS. Our approach starts from the update of the shell number and then picks out a candidate cutting contour C_i from the list of cutting contours. If the candidate has the authority of upward traversal or downward traversal, then starting from C_i, a new shell traversal is run along a certain direction (we set the rule of *first downward last upward*). If a crossing triangle is met that is cut by the cutting contour C_j during traversal, the authority of C_j may need to be updated, and it depends on two situations. One is that the cutting contour C_j and C_i belong to the same saddle; then, the authority of C_j for traversal along the current traversal direction is disabled. That is, if the current direction is downward, then the downward traversal authority of C_j is disabled and vice versa. The other situation occurs if the cutting contour C_j and C_i belong to different saddles; then, the authority of C_j for traversal along the opposite direction of the current is disabled (*e.g.*, if the current direction is downward, disable the upward traversal authority of C_j). The above operations are repeated until the bidirectional traversal authorities of all cutting contours are disabled.

Algorithm 2: Recursive traversal of current shell – traverseCurShell (D, T_i, C_i).

Input：

D – the traversal direction,

T_i – the triangle is to be visited,

C_i – the current cutting contour.

```
1:  if getType(T_i)=CrossingTriangle then
2:      visitFragments (D, T_i)
3:  else
4:      visitTriangle (T_i)
5:  end if
6:  TriangleSet S ← getAdjacentTriangles(T_i)
7:  for each triangle T in S do
8:      if getType (T)=CrossingTriangle then
9:          C_t ← getCriticalLoop(T)
10:         D_t ← D
11:         if getSaddle(C_i )≠getSaddle(C_t) then
12:             D_t ← ! D_t
13:         end if
14:         disableDirection (C_t, D_t)
15:     end if
16:     traverseCurShell (D, T, C_i)
17: end for
```

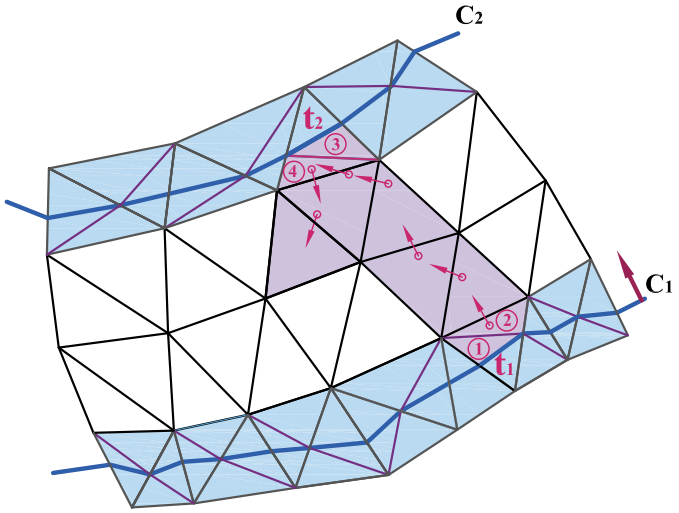


Fig. 6. One pass traversal that starts from C_1 along the upward direction.

As shown in Fig. 6, one pass traversal starts from a crossing triangle t_1, which is cut by the selected cutting contour C_1. Different visit ways are used to process different types of triangles. For the non-crossing triangle, it is directly marked with the current shell number. Otherwise, for the crossing triangle, its lower boundary fragment(s) or upper boundary fragment(s) with respect to the current traversal direction (*e.g.*, the upper boundary fragments of crossing triangle t_1 shown in Fig. 6) is visited, that is, its fragment(s) is marked with the current shell number. After visiting the triangle t, then a search for one of its adjacent triangle t_i with respect to the current traversal direction is performed. If t_i has not been visited, then recursively traverse from it.

Our strategy for shell traversal requires less computational cost. Because each triangle is visited only once, it takes $O(t)$-time in the step of shells traversal, where t is the number of triangles.

4.4 Construction of the Sub-cavity Network

Definition: The *sub-cavity network* (SCN) is a type of graph-like structure. It is abstracted as a weighted undirected graph $G(N, A, W)$ that consists of a set of nodes N, arcs A and weights W. The node n_i denotes the abstract representation of the sub-cavity s_i, an arc connects two adjacent sub-cavities, and a weight reflects the accessibility of surgical instruments from one sub-cavity to its adjacent one. We list the four properties of an SCN as follows.

(1) Each node of the SCN represents one sub-cavity. It can encode the quantitative geometric metrics for evaluating each sub-cavity, as well as the physical and physiological properties of the surrounding tissues. In our application, the major geometric metrics are composed of the *bottleneck index* and *directional compactness.*
(2) The weight w_{ij} encodes the *accessibility* of surgical instruments from the sub-cavity s_i to its adjacent one s_j.
(3) The SCN reflects the connectivity of sub-cavities. Once the surgery entrance has been selected, all candidate paths would be determined by the SCN, and the candidate paths would then be quantitatively evaluated using those parameters encoded in the nodes and arcs. For an example, as shown in Fig. 7, suppose n_1 is chosen for the surgical entrance, and the lesion is located inside the sub-cavity n_7, then there are four candidate paths for surgical tools, namely, n_1-n_2-n_4-n_5-n_7, n_1-n_2-n_4-n_6-n_7, n_1-n_3-n_4-n_5-n_7, and n_1-n_3-n_4-n_6-n_7.
(4) The SCN and Reeb graph can be converted into each other. As shown in Figs. 7 and 12, the number of nodes in an SCN is equal to that of arcs in a Reeb graph.

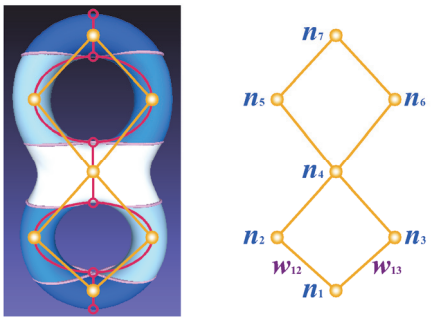


Fig. 7. The SCN (yellow) and the Reeb graph (red) of the double torus.

The algorithm for the construction of an SCN The construction method of an SCN on the basis of mesh segmentation will be presented in the rest of this section. An SCN can be built by turning an existing Reeb Graph of the same shape; however, this method is relatively complicated (*e.g.*, multiple saddle point need for special handling). We proposed an algorithm that can create an SCN directly. Our method requires only the adja-

cency information of shells, and critical points are not taken into account. Moreover, multiple saddle points or those saddles with the same function values do not require special processes (Fig. 8).

According to the definition of the SCN, the arcs number of the SCN is equal to that of the critical loops. Since the UBF and LBF (as shown in Fig. 4) of any *single-crossing face* (SCF, *i.e.*, the face only cut by one cutting contour) belong to two adjacent shells, for each critical loop L_i, take an SCF randomly, and then create an arc to connect the barycenters of these two shells. Repeat this step, until all the adjacent shells are connected by arcs. The procedure of constructing SCN is described in Algorithm 3.

Algorithm 3: Construction of SCN

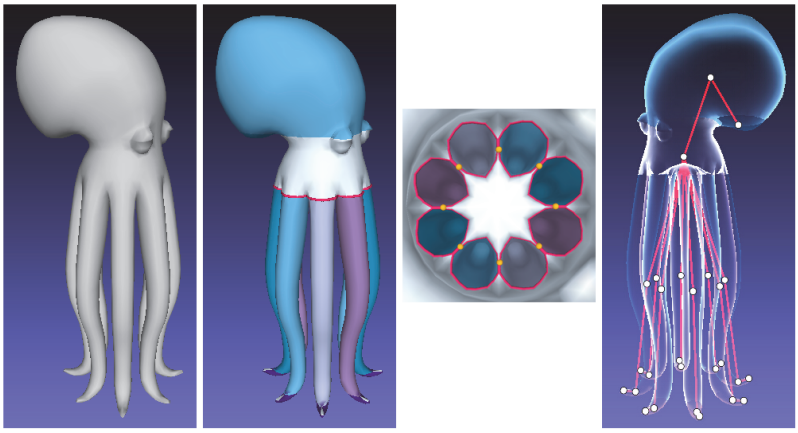
Input:

L – a set of critical loops,
B – a set of barycenters of shells,
$UBFs$ – a set of upper boundary fragments,
$LBFs$ – a set of lower boundary fragments.

Output:

N – a set of nodes, A – a set of arcs.

$N \leftarrow B$, $A \leftarrow \Phi$
for each critical loop $L_i \in L$ **do**
  $T_i \leftarrow$ an arbitrary SCF of L_i
  $UBF \leftarrow$ an UBF of T_i
  $LBF \leftarrow$ a LBF of T_i
  $S_1 \leftarrow$ shell# of UBF
  $S_2 \leftarrow$ shell# of LBF
  $e_1 \leftarrow$ barycenter of S_1
  $e_2 \leftarrow$ barycenter of S_2
  $A \leftarrow$ create arc a (e_1, e_2)
end for



(a) (b) (c) (d)

Fig. 8. The procedure of constructing an SCN; (a) the original mesh; (b) mesh segmentation; (c) the eight saddles with same function value; (d) the SCN of the given mesh.

5. CASE STUDIES: KNEE ARTICULATION

We used the articular cavity as an example to verify the proposed approach.

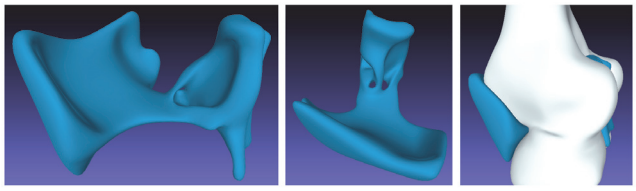
It is very difficult for surgical instruments to reach the lesion due to the knee joint cavity being very narrow. During knee arthroscopic surgery, surgeons need to constantly adjust the patient's lower limb to obtain the necessary operating space. Only an experienced doctor can be qualified for this job.

To solve this issue, we are working on a project with the help of orthopedic doctors. We first created the initial cavity model (Fig. 9) based on medical images; then, we simulated the tibio-femoral joint movement to create models (Fig. 10) of different poses. Next, we create the SCNs under different poses for the accessibility analysis to find out which pose and what type of instrument is best to access the designated position. In addition, we are designing a device that can automatically adjust the pose of the lower limb to assist in autonomous robotic surgery. The device is controlled by the computing results of accessibility.

Modeling Our construction process of the cavity model consisted of several steps, including medical image processing, 3D reconstruction, model smoothing, models assembling and automatic cavity model extraction. The medical images (CT and MRI) provided by our cooperative hospital were used for image segmentation and 3D reconstruction, and we then obtained the 3D mesh model of the tissues (*e.g.*, femur, tibia, cartilage, ligaments and menisci) that enclose the knee joint cavity. Automatic cavity model extraction was carried out after the assembly of the models. Tests of the intersection between normal and facets were utilized to distinguish the boundary facets of cavity, and these boundary facets were used as seeds to extract two inner cavity surfaces of the femur and the tibia by region growing. Then, the holes on the extracted surfaces were filled after the boundaries of the holes were detected automatically. A smoothing handle was also implemented to eliminate the jagged outline. Finally, the initial cavity model of the knee joint was created by suturing the two extracted surfaces.

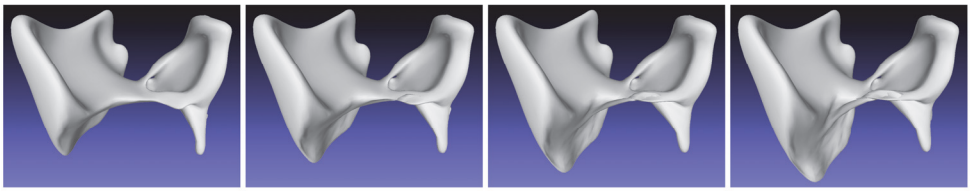
For those tissues that are completely surrounded by the cavity, such as the anterior cruciate ligament (ACL) and the posterior cruciate ligament (PCL), we subtracted them from the initial cavity model by Boolean operation. Holes were generated after subtraction. In this case, the knee articulation model includes three holes: one is related to the ACL, and the other two are related to the PCL (Figs. 9 (a) and (b)).

If the most common anterior portals are used in the knee arthroscopy surgery, surgical instruments are sometimes allowed to access the cavity through the infrapatellar fat pad, so we integrated the fat pad into the cavity model. A visual generalized cavity model that includes the infrapatellar fat pad is shown in Fig. 9.



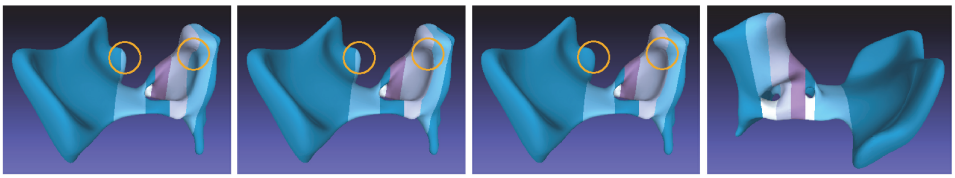
(a) (b) (c)

Fig. 9. The visual cavity model of the human knee joint; (a) the right side view; (b) the top view; (c) the cavity and its surrounding tissues.



(a) 15° (b) 30° (c) 45° (d) 60°

Fig. 10. The cavity model of the knee joint under different poses; the above angles represent the rotation of the tibia relative to the femur.



(a) (b) (c) (d)

Fig. 11. The result of segmentation; (a) before merging the small sub-cavities and combining the feature lines, (b) after combining the feature lines, (c) and (d) after merging and combining.

Results and discussion Fig. 11 shows the segmentation results by applying our segmentation algorithm. Only critical loops were used to subdivide the cavity in Fig. 11 (a), while the cutting contours combining the critical loops and the local feature contours of the constraint tissues were used in Fig. 11 (b). In Figs. 11 (c) and (d), the sub-cavities with smaller volume and that were not located at the border of the hole were merged into its adjacent sub-cavity.

The SCN constructed by our algorithm is shown in Fig. 11. We can see the number of simple rings (inside which there is no ring) in the SCN is with the same number of the genus in the cavity model, which shows that our segmentation method is able to identify the branch structures of the abstract shape.

As shown in Fig. 12, the cavity is divided into ten sub-cavities. Suppose n_1 is chosen for the surgical entrance, and the lesion is located inside the sub-cavity n_7; then, there are three candidate paths for surgical tools, namely, n_1-n_2-n_3-n_5-n_7, n_1-n_2-n_4-n_5-n_7 and n_1-n_2-n_6-n_7. To obtain an optimal path from the candidate paths, we also studied the accessibility of the cavity space; these research findings will be reported in the near future.

Robustness analysis To determine the stability of our algorithm, we will discuss whether the resolution of the input mesh will affect the results. Since the triangular mesh model is an approximation of the original cavity, the GPD may vary with different resolution. However, the angles between two GPDs are very small. As shown in Table 1, the average angle of all the six GPDs is 0.105°; in addition, the angles are not proportional to the difference of the triangle number. Although the variation of the GPD may change the number or positions of saddles (Table 2), our experiment showed that the consistent mesh segmentation and SCN could be obtained by redundant saddles filling automatically.

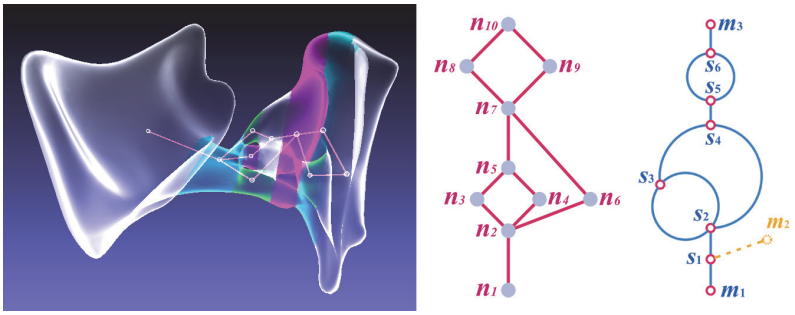


Fig. 12. The SCN (middle) and the Reeb graph (right) of the cavity. The dotted line indicates that the corresponding sub cavity has been merged.

Table 1. The GPDs affected by the number of triangles.

Number of triangles	The GPDs	The matrix of directional angles (°)					
74350	(–0.493360, –0.067566, 0.867196)	0	0.08	0.11	0.10	0.11	0.15
37174	(–0.493416, –0.067514, 0.867169)	0.08	0	0.09	0.07	0.08	0.13
18586	(–0.493487, –0.067532, 0.867126)	0.11	0.09	0	0.11	0.11	0.15
9292	(–0.493712, –0.067387, 0.867010)	0.10	0.07	0.11	0	0.08	0.12
4646	(–0.494068, –0.066952, 0.866841)	0.11	0.08	0.11	0.08	0	0.09
2322	(–0.494789, –0.066336, 0.866477)	0.15	0.13	0.15	0.12	0.09	0

Table 2. The critical points affected by the number of triangles.

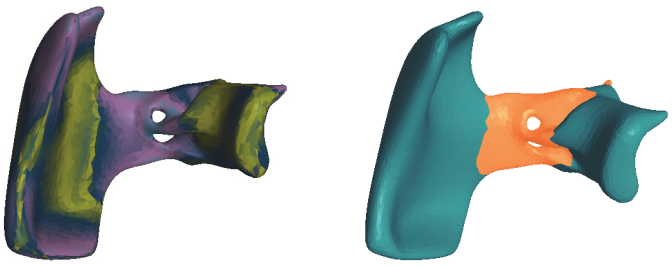
Number of triangles	Saddle	Maxima	Minima
74350	11	3	4
37174	10	3	3
18586	10	3	3
9292	10	4	2
4646	10	3	3
2322	10	3	3

6. COMPARISON

Segmentation Methods for shape segmentation depend on the specific application. Two requirements should be satisfied in our application context: one is a requirement for the semantic segmentation of an abstract shape; the other is requiring topological information.

However, the previous explicit boundary methods and the region-based methods are failure to our specific application since no topological information was taken into account in them. The former is not suitable for the subdivision of abstract shapes because they were designed to subdivide those objects either with clear hierarchy or with obvious contour between their functional components. For similar reasons, the latter cannot correctly distinguish the different topological branches. Fig. 13 illustrates the SDF values and the segmentation result by applying the approach in [12]. It shows that the difference

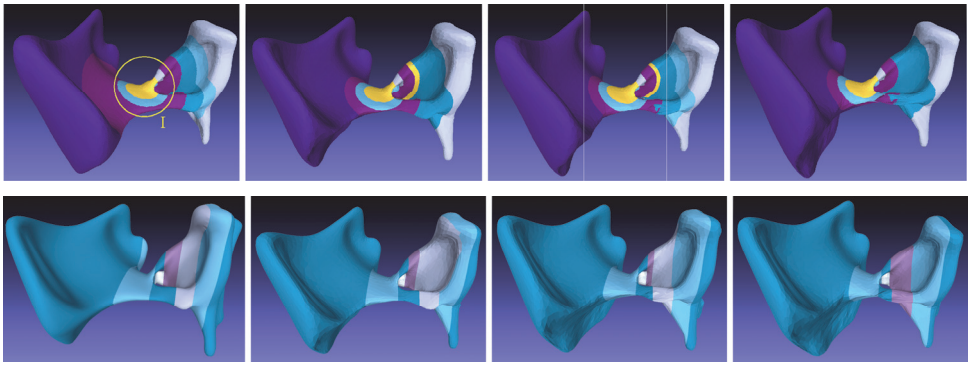
of SDF values is not obvious for the different branches, which lead to an incorrect segmentation (Fig. 13 (b)) by using clustering schemes.



(a) The SDF values. (b) The result of segmentation.

Fig. 13. The SDF values of the cavity (The source code is downloaded from http://www.cs.tau.ac.il/~liors/research/projects/sdf/).

The proposed algorithm can achieve approving semantic segmentation results for abstract shapes. Our method is topology-based. However, instead of using the common average geodesic distance, we defined a novel real function, which has been shown to be able to identify different branches of abstract shape. To compare our method with the geodesic distance based function, we validated the two algorithms by the cavity models under different poses. As shown in Fig. 14, the geodesic distance based method gained patches instead of the real subpart (inside the circle) due to its poor controllability of the segmentation contours, while the consistent segmentations are achieved by applying our algorithm.



(a) 0° (b) 15° (c) 30° (d) 45°

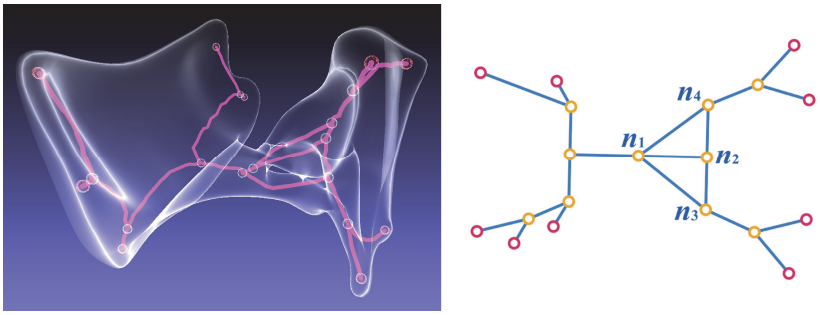
Fig. 14. The segmentation using a function with respect to geodesic distance (top row) and using our method (bottom row).

Shape descriptors We used the SCN as a shape descriptor instead of a skeleton [29], contour tree or Reeb graph. The reasons are as follows.

(1) In our application, the shape descriptor needed to include the abstraction of each sub-cavity to encode some information that is used to evaluate candidate paths. As men-

tioned in section 4.4, the information is with respect to each sub-cavity. However, the skeleton, contour tree and Reeb graph are incapable to encode the aforementioned information because these descriptors do not include the node with respect to the sub-cavity. As shown in Figs. 7 and 12, the node in the Reeb Graph represents the critical point. However, for the skeleton, the node represents end point or branch point, so it also cannot store the information of the sub-cavity.

(2) Although we can gain the SCN from the skeleton or the Reeb graph, the shape information is often abandoned, as only the abstract structure is left during the extraction of the skeleton and Reeb graph. As an exception, the skeleton in [29] can be converted to an SCN because it records the relation between original meshes and nodes. However, although it can obtain the skeleton with rich details, the algorithm based on mesh contraction in [29] may miss some topological rings due to the improper selection of parameters or excessive contraction. As shown in Fig. 15, only two simple rings ($n_1n_2n_3$ and $n_1n_2n_4$) are left in the skeleton, but the correct result should be three as illustrated in Fig. 12.



(a) The skeleton. (b) The simplified skeleton.

Fig. 15. The skeleton descriptor of the cavity model.

7. CONCLUSION AND FUTURE WORK

We have presented a feasible approach to subdivide abstract shapes into semantic parts via explicit boundary contour extraction. The cutting contours that are served as boundaries of parts are based on the mesh topology and local shape geometry. To compute the critical loops, we defined a real function with respect to the application context, and the values of function are invariant under translation, rotation and scaling transformation. Our method ensures that the segmentation result has semantic parts due to the controllable cutting contours. We also defined a graph-like structure representation, the SCN, to encode geometrical, topological and physiological information. We proposed an algorithm to create an SCN directly. Our method requires only the adjacency information of shells, and critical points are not taken into account. Moreover, multiple saddle points do not require the use of special process.

Several aspects of the proposed approach can be improved in future research. First, a function containing topological information and geometrical information, instead of computing critical loops and feature contours separately, should be constructed. Second,

the problem of real-time segmentation when the shape of the body cavity is time-varying should be addressed. Third, the sub-cavities should be further subdivided using geometrical and physiological information. Fourth, after completing the test of the assisted device and improving the software system, we will verify our method in minimally invasive orthopedic surgery.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable comments. This work was supported in part by the National Natural Science Foundation of China (Grant No. 51275094, 61300106), by the Science and Technology Program of Guangdong Province (Grant No. 2016A040403108, 2015A030310112, 2014B040401012), by Science and Technology Planning Project of Guangdong Province (Grant No. 2017B020210009), by the Guangdong Provincial Key Laboratory of High Performance Computing (Grant No. TH1526), and by Research Fund for Young Scholars of GDUT (Grant No. 14ZK0022). We would like to thank the No. 3 Affiliated Hospital of Guangzhou University of Chinese Medicine, for providing medical images and guidance of medical knowledge.

REFERENCES

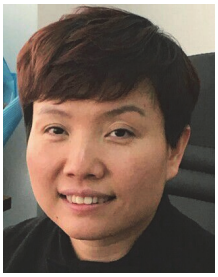
1. E. de Momi, C. Caborni, F. Cardinale, L. Castana, G. Casaceli, M. Cossu, L. Antiga, and G. Ferrigno, "Automatic trajectory planner for StereoElectroEncephaloGraphy procedures: A retrospective study," *IEEE Transactions on Biomedical Engineering*, Vol. 60, 2013, pp. 986-993.
2. E. de Momi, C. Caborni, F. Cardinale, G. Casaceli, L. Castana, M. Cossu, R. Mai, F. Gozzo, S. Francione, L. Tassi, G. Lo Russo, L. Antiga, and G. Ferrigno, "Multi-trajectories automatic planner for StereoElectroEncephaloGraphy (SEEG)," *International Journal of Computer Assisted Radiology and Surgery*, Vol. 9, 2014, pp. 1087-1097.
3. A. Schoob, D. Kundrat, L. Kleingrothe, L. Kahrs, N. Andreff, and T. Ortmaier, "Tissue surface information for intraoperative incision planning and focus adjustment in laser surgery," *International Journal of Computer Assisted Radiology and Surgery*, Vol. 10, 2015, pp. 171-181.
4. A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, Vol. 27, 2008, pp. 1539-1556.
5. A. Golovinskiy and T. Funkhouser, "Consistent segmentation of 3D models," *Computers and Graphics*, Vol. 33, 2009, pp. 262-269.
6. E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3D mesh segmentation and labeling," *ACM Transactions on Graphics*, Vol. 29, 2010, pp. 1-12.
7. K. Xu, H. Li, H. Zhang, D. Cohen-Or, Y. Xiong, and Z.-Q. Cheng, "Style-content separation by anisotropic part scales," *ACM Transactions on Graphics*, Vol. 29, 2010, pp. 1-9.
8. Q. Huang, V. Koltun, and L. Guibas, "Joint shape segmentation with linear programming," *ACM Transactions on Graphics*, Vol. 30, 2011, pp. 1-12.

9. O. Sidi, O. Van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or, "Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering," *ACM Transactions on Graphics*, Vol. 30, 2011, pp. 1-9.
10. M. Meng, J. Xia, J. Luo, and Y. He, "Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization," *Computer-Aided Design*, Vol. 45, 2013, pp. 312-320.
11. S.-K. Wong, J.-A. Yang, T.-C. Ho, and J.-H. Chuang, "A skeleton-based approach for consistent segmentation transfer," *Journal of Information Science and Engineering*, Vol. 30, 2014, pp. 1053-1070.
12. L. Shapira, A. Shamir, and D. Cohen-Or, "Consistent mesh partitioning and skeletonisation using the shape diameter function," *The Visual Computer*, Vol. 24, 2008, pp. 249-259.
13. Y. Lee, S. Lee, A. Shamir, D. Cohen-Or, and H.-P. Seidel, "Mesh scissoring with minima rule and part salience," *Computer Aided Geometric Design*, Vol. 22, 2005, pp. 444-465.
14. A. Golovinskiy and T. Funkhouser, "Randomized cuts for 3D mesh analysis," *ACM Transactions on Graphics*, Vol. 27, 2008, pp. 1-12.
15. O. K.-C. Au, Y. Zheng, M. Chen, P. Xu, and C.-L. Tai, "Mesh segmentation with concavity-aware fields," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 18, 2012, pp. 1125-1134.
16. Y. Zheng, C.-L. Tai, and O. K.-C. Au, "Dot scissor: A single-click interface for mesh segmentation," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 18, 2012, pp. 1304-1312.
17. R. Liu, H. Zhang, A. Shamir, and D. Cohen-Or, "A part-aware surface metric for shape analysis," *Computer Graphics Forum*, Vol. 28, 2009, pp. 397-406.
18. T.-C. Ho and J.-H. Chuang, "Volume based mesh segmentation," *Journal of Information Science and Engineering*, Vol. 28, 2012, pp. 705-722.
19. O. van Kaick, N. Fish, Y. Kleiman, S. Asafi, and D. Cohen-Or, "Shape segmentation by approximate convexity analysis," *ACM Transactions on Graphics*, Vol. 34, 2014, pp. 1-11.
20. X. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3D mesh segmentation," *ACM Transactions on Graphics*, Vol. 28, 2009, pp. 1-12.
21. S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno, "Reeb graphs for shape analysis and applications," *Theoretical Computer Science*, Vol. 392, 2008, pp. 5-22.
22. S. Berretti, A. D. Bimbo, and P. Pala, "3D mesh decomposition using Reeb graphs," *Image and Vision Computing*, Vol. 27, 2009, pp. 1540-1554.
23. J. Tierny, J.-P. Vandeborre, and M. Daoudi, "Topology driven 3D mesh hierarchical segmentation," in *Proceedings of International Conference on Shape Modeling*, 2007, pp. 215-220.
24. G. Patanè, M. Spagnuolo, and B. Falcidieno, "A minimal contouring approach to the computation of the Reeb graph," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 15, 2009, pp. 583-595.
25. M. Attene, F. Robbiano, M. Spagnuolo, and B. Falcidieno, "Characterization of 3D shape parts for semantic annotation," *Computer-Aided Design*, Vol. 41, 2009, pp. 756-763.
26. X. Ni, M. Garland, and J. C. Hart, "Fair morse functions for extracting the topologi-

cal structure of a surface mesh," *ACM Transactions on Graphics*, Vol. 23, 2004, pp. 613-622.
27. D. V. Vranic, "3D model retrieval," Ph.D. Thesis, Institute of Information, University of Leipzig, 2004.
28. V. B. Sunil and S. S. Pande, "Automatic recognition of features from freeform surface CAD models," *Computer-Aided Design*, Vol. 40, 2008, pp. 502-517.
29. O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee, "Skeleton extraction by mesh contraction," *ACM Transactions on Graphics*, Vol. 27, 2008, pp. 1-10.

Jian-Qing Mo (莫建清) received his M.S. degree from Chongqing University, China. He is currently a Lecturer and Ph.D. candidate of Guangdong University of Technology. His research interests include virtual reality, computer graphics and computer assisted surgery.

Han-Wu He (何漢武) received his Ph.D. degree in Electromechanical Engineering in 2001, Master degree in Manufacturing Automation Engineering in 1992 and Bachelor degree in Mechanical Engineering in 1988, all from Huazhong University of Science and Technology at Wuhan, China. He is currently a Professor of the School of Electromechanical Engineering at Guangdong University of Technology (GDUT) at Guangzhou, China. He is also the head of the virtual reality research group at GDUT. His main research interests are virtual reality, augmented reality, and applications in product design, digital smart factory, manufacturing systems and medical treatment. His research projects have been sponsored by a number of organizations including the Natural Science Foundation of China, the Natural Science Foundation of Guangdong Province and Department of Science and Technology of Guangdong Province. He received the third prize for the National Science and Technology Progress from China in 1995.



Jin-Fang Li (李晉芳) received her Ph.D. degree from Beihang University in 2004. She is currently an Associate Professor of Guangdong University of Technology, China. Her research interests include virtual reality, computer assisted surgery and computer-aided design.

Yu-Wei Wei (韋宇煒) received his M.S. degree in Computer Science from Guangdong University of Technology in 2005. He is currently a Lecturer of Guangdong University of Technology. His research interests include algorithm design and analysis, surface electromyograph analysis, and software structure design.