Pacific Graphics 2010
P. Alliez, K. Bala, and K. Zhou
(Guest Editors)

Volume 29 (2010), Number 7

Contour-based Interface for Refining Volume Segmentation

Takashi Ijiri and Hideo Yokota
RIKEN

Abstract

Medical volume images contain ambiguous and low-contrast boundaries around which existing fully- or semi-automatic segmentation algorithms often cause errors. In this paper, we propose a novel system for intuitively and efficiently refining medical volume segmentation by modifying multiple curved contours. Starting with segmentation data obtained using any existing algorithm, the user places a three-dimensional curved cross-section and contours of the foreground region by drawing a cut stroke, and then modifies the contours referring to the cross-section. The modified contours are used as constraints for deforming a boundary surface that envelops the foreground region, and the region is updated by that deformed boundary. Our surface deformation algorithm seamlessly integrates detail-preserving and curvature-diffusing methods to keep important detail boundary features intact while obtaining smooth surfaces around unimportant boundary regions. Our system supports topological manipulations as well as contour shape modifications. We illustrate the feasibility of our system by providing examples of its application to the segmentation of bones, muscles, kidneys with blood vessels, and bowels.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Methodology and Techniques]: Interaction techniques. I.4.6 [Segmentation].

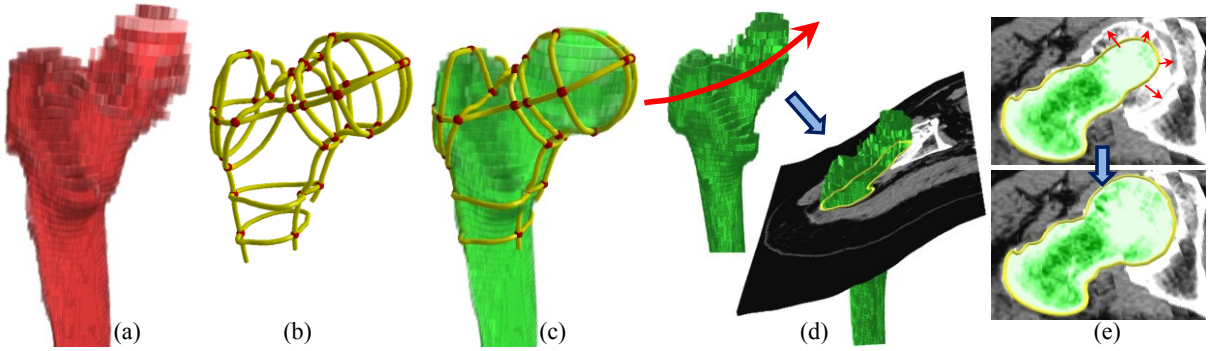

Figure 1. *Refining the end of a thigh bone with our system. Starting with segmentation data containing errors (a), the user places and modifies multiple contour curves (b) to refine the segmented region with voxel-level accuracy (c). Our system provides a set of user interfaces for the placement of contour curves (d) and their modification (e).*

1. Introduction

Volume segmentation is the process of separating volume data into semantic units. It is a fundamental process for obtaining useful information from volume data including shape, topology, and various other measurements. The segmented data are also useful in constructing models for physical and interactive simulation. Recent developments in medical imaging devices, such as magnetic resonance imaging and computed tomography (CT), increase the importance of efficient and intuitive tools for three-dimensional (3D) volume image segmentation.

Traditional volume segmentation systems require the user to specify the foreground region in each two-dimensional (2D) slice. Such systems accurately reflect the user's intention, even though the segmentation process is very time consuming. Many fully- and semi-automated systems have been proposed to make the segmentation process efficient. However, as Owada et al. [ONT05] have pointed out, no fully automated systems are yet available because segmentation remains dependent on the user's subjective interpretation, which is impossible to obtain without user intervention. Semi-automated systems allow users to include subjective information and usually determine a boundary based on user-provided information and

low-level features of the volume image (e.g., colors, gradient magnitudes, or edges). For example, region growing, one of the most popular semi-automatic methods, starts from user-provided seeds and inflates an object region considering local colors and gradients of the image. Semi-automatic systems involve a fair amount of user interaction, but still cannot eliminate local errors around the ambiguous and low-contrast boundaries that often appear in medical volume images. Achieving fully satisfactory segmentation with voxel-level accuracy remains difficult. Therefore, we believe that it is very important to provide an efficient framework that allows users to refine segmentation results of existing automatic systems.

This paper proposes a novel system for refining already extracted region data by modifying 3D contours. The session begins with segmented data obtained using any existing method (Fig. 1(a)). In this work, we used graph cut [BVZ01; LSTS04] for the initial segmentation. The user first draws a cut stroke on a screen to place a curved cross-section and a bounding contour (Fig. 1(d)). Then the user modifies the shape of the contour so that it traces a plausible boundary (Fig. 1(e)). The influence of the contour modification is propagated in the volumetric region through a boundary surface deformation; we deform a boundary surface of the foreground region using the modified contours as constraints and update the region inside the deformed boundary. Our system can deal with non-planar or open contour constraints. We support topological contour modifications as well as contour shape deformations. The modified contours can be registered as constraints for the subsequent process. The user repeats the contour construction, modification, and registration cycle until all undesired boundaries disappear (Figs. 1(c)).

With our system, the user can easily place contours where only human but automatic algorithms can perceive. The number of required contour modifications is relatively small because they influence the volumetric region through boundary surface deformation. Our boundary surface deformation algorithm maintains the detail features of important boundary shapes and achieves a smooth surface around unimportant boundary regions by combining Laplacian-preserving and curvature-diffusing methods.

2. Related Work

Our work builds on several existing techniques from different fields. We briefly review representative work from the fields of image segmentation and surface deformation.

Image segmentation. Two-dimensional image segmentation has received much attention over the past decades, resulting in many different approaches such as thresholding, k-mean clustering, active contours, and graph cut algorithms, which have been the subject of at least one survey [PXP98]. We will explain several approaches that are closely related to our work. Active contour was originally presented by Kass et al. [KWT88] and many extensions have been published [HPE*08]. In that method, a contour curve is deformed so as to minimise two types of energy: internal energy that is sensitive to the contour shape and external energy that is sensitive to the local image features under the contour. A graph cut algorithm was used for image segmentation by Boykov et al. [BVZ01], and Li et al. [LSTS04] extended it to provide an interactive image cut-out tool. These systems require the user only to mark foreground and background regions by drawing rough strokes. The systems construct an energy function based on user specification and then minimise it using the graph cut algorithm.

Segmentation algorithms primarily developed for 2D are often applicable to 3D with minor modifications. Providing an intuitive interface is important when developing 3D image segmentation tools. Some systems allow users to interact with cross-sectional planes to specify foreground and background regions in 3D space [TLM03; WBC*05; SHN03]. Others support drawing strokes directly on the volume rendered screen [ONT05; YZNC05] where the user input is then passed to various segmentation algorithms.

Note that segmentation algorithms usually rely on low-level features of the volume image. Hence, they often fail around the low-contrast areas or regions with ambiguous boundaries that frequently appear in medical images.

Contour-based mesh construction for volume segmentation. Some researchers have used mesh construction techniques in volume segmentation procedures. The SketchSurfaces system [AM07] allows the user to draw closed contours on parallel cross-sectional planes in volume images. From these contours, the quick-hull algorithm constructs a closed surface, which is then used as an initial shape in the active contour method. VolumerViewer [SLJ*09] also constructs a closed boundary surface from user-drawn oblique contours using an algorithm introduced by Liu et al. [LBD*08] and the voxels inside the surface are segmented as foreground. These systems produce intuitive contouring interfaces by integrating mesh construction algorithms. However, they were not designed for the refinement process of regions that have already been segmented. In addition, these systems are limited to closed and planar contours. Our system, on the other hand, works with open and non-planar contours, permitting efficient local modification and effective manipulation of curved objects.

Laplacian surface deformation. Our scheme for refining the foreground region is based on a surface deformation method called Laplacian surface deformation [Sor06; BS08]. It considers mesh deformation as an energy-minimization problem. The energy function contains the terms for detail preservation and positional constraints. Based on this technique, Nealen et al. [NSAC05] presented a set of sketching gestures to specify constraints. The FiberMesh system [NISA07], which strongly influenced our system, provides a sketch-based 3D modelling interface that supports both the initial creation and subsequent deformation processes. During deformation, the user can freely add curved fiber constraints and modify the fiber as a handle for surface deformation. To the best of our knowledge, these Laplacian surface deformations have never before been used for volume segmentation purposes.

3. System Overview

Our system efficiently and intuitively refines segmented regions by modifying multiple curved contours. The input includes a 3D volume image and foreground region data (i.e., a binary mask), which may contain errors. Even though we rely on 3D graph cut [BVZ01; LSTS04] in this paper for initial segmentation, many other segmentation techniques are available.

Before starting, the system constructs a boundary surface enveloping the initial foreground region using the Marching Cubes algorithm [LC87]. In the refinement process, the system first deforms this surface based on the user-modified contour constraints and updates the region inside the surface. The typical user's refinement process includes the following steps: observing a current foreground region (Fig. 1(a)), cutting the foreground region to construct a cross-section and contours (Fig. 1(d)), and modifying the shapes of the contours (Fig. 1(e)). The modified contours can be registered as constraints. The user repeats these steps until satisfied with the updated region. Our system gives the user the option of running a graph cut algorithm at the voxel level to fit a region boundary to a high-contrast edge [LSTS04].

In some sense, our system can be seen as a 3D extension of the boundary editing tool in the Lazy Snapping system [LSTS04]. It allows the user to refine 2D segmented data with pixel accuracy by directly modifying the bounding polyline. However, extending this to 3D is not straightforward. For example, simply allowing the user to modify a 2D bounding polyline on each slice is tedious; our system provides a solution to this problem by retaining efficiency for 3D situations while maintaining a user interface similar to that proposed by Li et al. [LSTS04].

4. User Interface

Visualisation. Efficient visualisation of the current foreground region is very important for supporting the user in detecting suspicious boundary parts that need to be fixed. As shown in Figure 2, the system provides three visualisation modes: surface rendering, binary volume rendering, and surface rendering with 3D volume texture. Surface and volume rendering helps the user to observe the shape of the region, which provides significant information. Textured surface rendering allows one to examine the volume image on the boundary surface. Because the boundary color of an object is usually isotropic, at least locally, a boundary that exhibits significant changes in color likely contains errors. Our textured surface rendering helps the detection of such suspicious boundary regions.

Cut stroke. The user draws a cut stroke running across the suspicious boundary regions on the screen to construct a curved cross-sectional surface and contours. The cross-sectional surface is constructed by sweeping the user-drawn 2D cut stroke along the depth direction. Contour curves are constructed by computing intersections between all edges of the boundary surface and the cross-sectional surface, and then aligning the crossing points. Thus, a contour curve C is a polyline, each vertex $\{\mathbf{p}_i\}$ of which is on an edge ($\mathbf{v}_k$, $\mathbf{v}_l$) of the boundary surface:

$$C = \{\, \mathbf{p}_i \mid \mathbf{p}_i = t\mathbf{v}_k + (1-t)\mathbf{v}_l,\ \ t \in [0,1] \,\}.$$

Figures 1(d) and 3 demonstrate the effects of the cut stroke. If the user draws a cut stroke along a curved object such as a bowel, the system generates a contour that captures the curved features of the object very well (Fig. 3(a)). A cut stroke running across the object multiple times results in multiple contours (Fig. 3(b)). When a cut stroke starts from the outside and ends inside the object, the system returns an open contour curve that is useful for local shape editing (Fig. 3(c)). To support an efficient 3D navigation, our system provides an auto-view-transfer option in which the camera immediately *flies* in front of the generated cross section after drawing a cut-stroke.

After constructing a contour curve, the system detects crossing points between the newly constructed contour and already stored contours. This detection is performed on the 2D screen plane; the system projects contour curves onto the screen and checks the existence of intersections between the curves. If crossing points exist, we subdivide intersecting line segments, back-project the detected crossing points, and place red spheres at the points (Fig. 3(d)). These crossing points are fixed in the subsequent process.

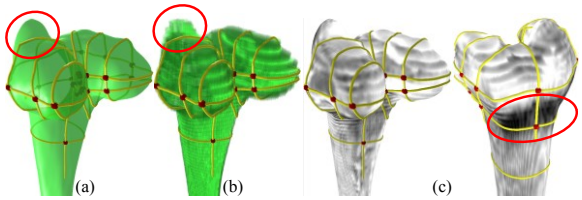

Figure 2. *Three rendering modes for the knee section of a thigh bone. While surface (a) and volume (b) rendering display suspicious shapes in the region, the textured surface (c) clearly illustrates significant changes in boundary color (red circle).*

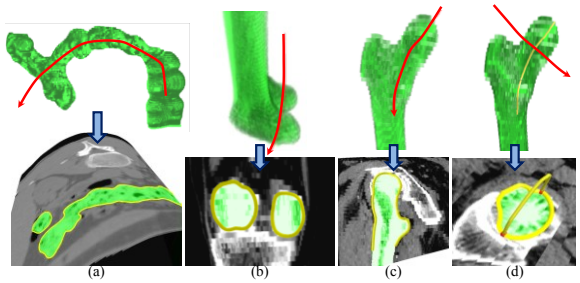

Figure 3. *Effects of cut-strokes.*

Contour deformation. After setting contours, the user deforms their shape so that they trace plausible boundaries on the cross-sectional surface. Our system provides five tools for this purpose: drawing, point-dragging, smooth-rubbing, snake-rubbing, and contour-deletion tools (Fig. 4).

The drawing tool allows the user to modify a part of the contour by drawing its desired shape on the cross section (Fig. 4(a)). The point-dragging tool provides a similar interface to that of fiber deformation described by Nealen et al. [NISA07]. It allows the user to grab a contour at any point

and pull it to the desired location (Fig. 4(b)). Deformation of the contour preserves local details as much as possible. The smooth-rubbing tool has the same interface as the rubbing tool of Nealen et al. [NISA07]; if the user drags the cursor back and forth in a rubbing motion on a part of the contour, the rubbed region becomes smooth (Fig. 4(c)). While the snake-rubbing tool has the same interface as smooth-rubbing, the rubbed region moves along the gradient direction to fit edges like the snake [KWT88; HPE*08]. Note that snake-rubbing works well only when the target cross-section has high contrast edges as shown in Figure 4(d). Finally, the contour-deletion tool allows the user to delete undesired contours when multiple contours exist (see Section 5.2 in detail). During contour modification, the crossing points highlighted with red spheres are fixed, except in the case where the user explicitly drags them using the point-dragging tool. Contour deformation is limited to the cross-sectional surface.

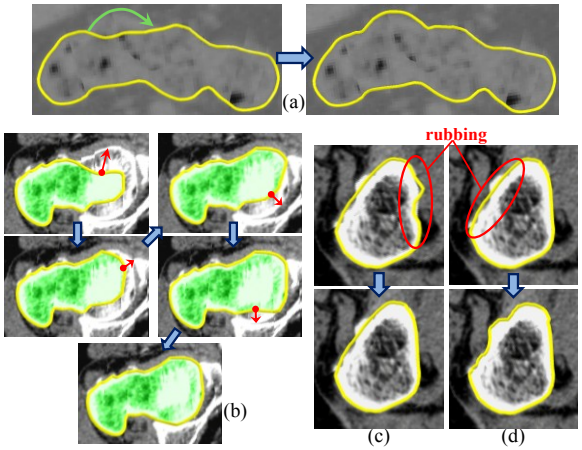

Figure 4. *Contour manipulations using the drawing tool(a) point-dragging tool(b), smooth-rubbing tool(c), and snake-rubbing tool(d).*

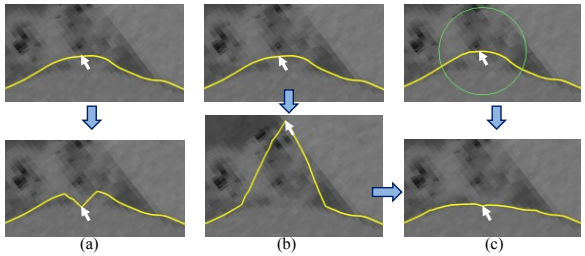

Figure 5. *Contour ROI Definition. With the original peeling interface(a)[IMH05], the user temporally drags a point far from the picked position to expand the ROI (b), and then releases it at the desired location. We display a circle around the cursor and it determines the initial ROI (c).*

In the original FiberMesh [NISA07], the fiber deformation tool (point-dragging in this paper) determines the deformed region of the fiber (region of interest, ROI) using a peeling interface [IMH05]; starting from a single point, it expands the ROI proportional to the amount of pulling. Our preliminary user study indicates that this peeling is difficult for novice users. The peeling interface forces the user to perform two tasks in a single dragging operation: ROI determination and point placement. We found that almost all users simply drag the point to the desired position without considering the ROI resulting in unwanted spiky contours (Fig. 5(a)). We added a simple extension to overcome this problem. Around the cursor, we display a circle whose radius can be controlled using the mouse wheel. We then determine all contour segments in the circle to be the initial ROI. This simple extension successfully prevents novice users from generating undesirable spiky shapes (Fig. 5(c)).

5. Enveloping Surface Deformation

We provide two surface-deformation algorithms to implement the user interface described in Section 4.1. The first one deforms the boundary surface to satisfy the user-specified contour constraints. The second modifies the surface topology so that it satisfies the new contour layout.

5.1 Surface Deformation by Contour Modification

Two types of contours exist in our setup: those that have already been modified and registered, and those that are currently being modified (i.e., active contours). The surface deformation is localised to a part of the surface near the active contours (ROI). The ROI is defined as an *N*-ring neighborhood of the surface edges constrained by the active contours, where *N* is a user's parameter (Fig. 6). Notice that since the boundary surface is generated by the Marching Cubes algorithm and the vertices are almost uniformly distributed. Thus, the N-ring neighborhood of the active contour generates a fair ROI in practice.

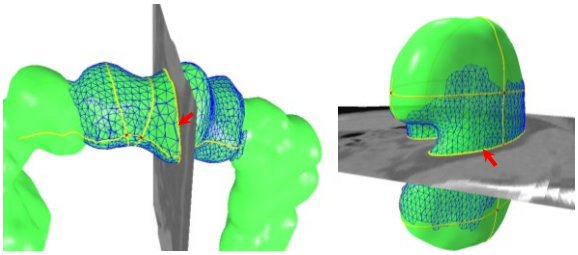

Figure 6. *ROI (blue mesh) is defined as an N-ring neighborhood of an active contour marked by a red arrow.*

Constraint-based surface deformation has been the subject of active research. One of the most well-known methods is Laplacian surface editing [SLC*04], which deforms a surface while preserving both the user constraints and the geometric details as much as possible. This method encodes geometric detail at each vertex using Laplacian coordinates and updates the shape of the surface by solving the following minimization problem:

$$\underset{\mathbf{v}_i}{\operatorname{argmin}} \left\{ \sum_i ||\mathcal{L}(\mathbf{v}_i) - \boldsymbol{\delta}_i^{init}||^2 + \sum_i ||\mathbf{v}_i - \mathbf{v}_i^0||^2 \right\} \ldots (1)$$

where $\mathcal{L}(\cdot)$ is a discrete graph Laplacian operator, $\boldsymbol{\delta}_i^{init}$ is a Laplacian vector of the initial surface at the ith vertex $\mathbf{v}_i$, and $\mathbf{v}_i^0$ is a constraint position of $\mathbf{v}_i$. This quadratic energy-minimization problem results in a sparse linear system. While Laplacian surface editing preserves local details, the

FiberMesh system achieves a smooth surface in which the Laplacian magnitude (LM, i.e., the approximation of scalar mean curvature) is smoothly distributed. FiberMesh includes a two-step process. It first smoothly distributes LMs $\{c_i\}$ to all vertices by solving the following optimization,

$$\underset{c_i}{\operatorname{argmin}} \left\{ \sum_i ||\mathcal{L}(c_i)||^2 + \sum_i ||c_i - c_i'||^2 \right\} \ldots (2)$$

where the first term requires that the neighboring LMs vary smoothly and the second term requires the LMs at all vertices to be near the current LMs. After obtaining the LM distribution, FiberMesh estimates a Laplacian vector for each vertex as, $\boldsymbol{\delta}_i^{LM} = c_i \times \mathrm{A}_i \times \mathbf{n}_i$ and solves the optimization of Eq. (1) by using $\boldsymbol{\delta}_i^{LM}$ as the target Laplacian. The A_i is an area estimate and $\mathbf{n}_i$ is the normal of $\mathbf{v}_i$. FiberMesh repeats solving Eqs. (2) and (1) until convergence. While this algorithm requires solving a sparse linear system several times, the expensive matrix factorization is required only once at the beginning.

We would like to apply these two techniques to our situation; however, selecting the appropriate one is difficult. In our system, the surface represents a boundary shape of the initial (or current) foreground region. In a surface section around contours with little or no deformation, the local shape is important and geometric details should be preserved. In contrast, in a surface region around highly deformed contours, the local shape is no longer important and a smooth surface is preferred.

For this reason, we combine detail-preserving and LM-diffusing surface deformations by switching between them seamlessly for regions around slightly deformed and highly deformed contours. To do this, we first diffuse displacement amounts $\{d_{pj}\}$ of contour's vertices $\{\mathbf{p}_j\}$ over the all vertices of the boundary surface $\{\mathbf{v}_i\}$, similarly to the LM diffusion in Eq. (2) as

$$\underset{d_i}{\operatorname{argmin}} \left\{ \sum_i ||\mathcal{L}(d_i)||^2 + \sum_j \left(||d_k - d_{pj}||^2 + ||d_l - d_{pj}||^2 \right) \right\} \quad \ldots (3)$$

where d_i is the diffused displacement amount at $\mathbf{v}_i$. While the first term requires that the neighboring displacement amounts vary smoothly, the second term constrains the displacement amounts at each pair of surface vertices which correspond to a contour's vertex. Note that each contour vertex $\mathbf{p}_j$ is on an edge $(\mathbf{v}_k, \mathbf{v}_l)$ of the surface.

Next, we compute a new vertex position by solving the following least-square system:

$$\underset{\mathbf{v}_i}{\operatorname{argmin}} \left\{ \sum_i \left(a_i ||\mathcal{L}(\mathbf{v}_i) - \boldsymbol{\delta}_i^{init}||^2 + (1 - a_i)||\mathcal{L}(\mathbf{v}_i) - \boldsymbol{\delta}_i^{LM}||^2 \right) + \sum_j ||t\mathbf{v}_k + (1-t)\mathbf{v}_l - \mathbf{p}_j||^2 \right\} \ldots (4)$$

While the first term requires that each vertex Laplacian be close to the initial Laplacian (i.e., detail preservation), the second term requires that each Laplacian be close to the Laplacian integrated from diffused LMs (i.e., LM diffusion). A set of scalar values $a_i \in [0,1]$ is calculated by $a_i = \frac{d_i}{kr}\ (d_i < kr),\ \ a_i = 1 (d_i > kr)$,, where k is a global parameter that we set to 5.0 and r is the voxel edge length. The third term places point constraints on edges. The point constraint on an edge is defined with one parameter t, $\mathbf{p}_j = t\mathbf{v}_k + (1-t)\mathbf{v}_l$ [NSAC05]. This hybrid method successfully combines the advantages of the Laplacian-preserving and LM-diffusing methods (see Section 6 in detail).

Note that a single contour modification does not guarantee that surface regions far from the contour are appropriately located along the boundary of the volume image. To address this issue, the SketchSurface system adjusts the surface by using the image information (i.e., active contour method). However, in our practical observation, surface regions that requires modification usually located around low-contrast or ambiguous areas of the image, since the initial segmentation algorithm fails around such regions. Image-based methods are difficult to apply to such regions. Instead, we simply allow the user to iteratively place and modify next contours until wrong surface region disappears.

5.2 Surface topology modification

When the user deletes a contour using the contour deletion tool, the topology of the surface must be changed to satisfy the new contour layout. Our system deals with two layout types: a parallel layout in which several closed contours are arranged in parallel, and a nested layout in which two contours are nested (Fig. 7). The system does not allow the deletion of open contours or the outer contour of the nested layout because the resulting boundary surface would be unpredictable and such deletion is practically unnecessary.

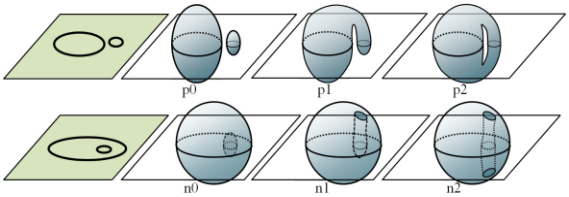

Figure 7. *Surface topology types for a parallel layout (top) and nested layout (bottom). The same contour layout is constructed with two isolated closed surfaces (p0, n0), a single bent closed surface homeomorphic to a sphere (p1, n1), or a surface homeomorphic to a torus (p2, n2).*

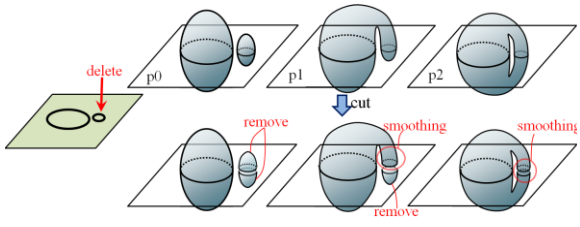

Figure 8. *Surface topology modification for p0, p1, and p2. The system cuts the surface at the target contour, removes isolated surface parts, and smoothes the remaining cross-sections. The same process is applicable to n0, n1, and n2.*

We do not deal with the layout in which more than two contours are nested because such layouts rarely appear in practice.

When two closed contours are placed in parallel or are nested, three possible boundary surface topology types exist for each (Fig. 7). When the shorter contour is deleted, we modify the surface topology. Figure 8 shows how our system manages six topology types using the same process. First it cuts the surface at the target contour and fills the holes in the cross-section. In types *p0* and *n0*, this process splits a closed surface into two parts. In types *p1* and *n1*, a closed surface is separated out. In types *p2* and *n2*, no new isolated part is generated. Then the system checks whether the separated surface parts have contour constraints. If not, it simply removes the separated part. Finally, it smoothes the remaining cross-sections.

6. Results and Discussion

We tested our prototype system on a 2.93-GHz Intel Core i7 CPU. Our system provides an interactive environment to the user: after each action, it updates the boundary surface and foreground region in real time and returns immediate feedback.

Figure 9 illustrates the feasibility of our surface deformation algorithm. The user refines thigh bone data (Fig 9(a)) by placing and modifying contours (Fig 9(b)). In this example, only the region at the end (head) of the bone contains errors. The contours around the head are modified significantly while those in the other region retain their initial shape. In this situation, the user wants to have a smooth surface around the head and keep the sharp features

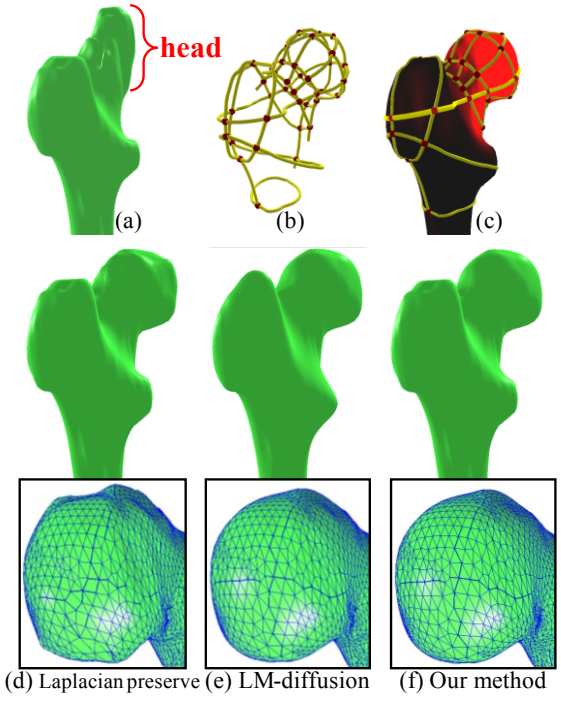

Figure 9. *Surface deformation using three algorithms.*

(initial shape) intact in the other region. However, Laplacian preservation keeps the sharp features for both the head and the other region (Fig 9(d)). On the other hand, LM diffusion returns a smooth surface for both regions resulting in the loss of important sharp features (Fig 9(e)). Our algorithm combines the best of both methods, obtaining a smooth surface around the head while maintaining important detail features in the other region (Fig 9(f)). Figure 9(c) shows the distribution of a_i value; the values 0 and 1.0 correspond to the colors black and red. In the head region with large a_i values (red), the LM diffusion effect predominates, resulting in a smooth surface. In the region with small a_i values (black), the Laplacian-preservation is dominant and sharp features are maintained.

Figures 1 and 10 show the results of the thigh bone segmentation which was initially segmented with a 3D graph cut and subsequently refined with our system. In a CT image, a thigh bone has a high-contrast boundary in the shaft region and no refinement is required. However, on the top and bottom, cancellous (low intensity) bone is covered with thin cortical (high intensity) bone, making accurate region segmentation difficult for automatic algorithms. With our system, the user can refine the two sides of the thigh bone by placing and modifying about 10 contours in less than 15 minutes for each (Figs. 1 and 10). In this thigh bone example, we found a hole (surface topology type *n2*) that our contour-deletion tool successfully removed (Fig. 10left).

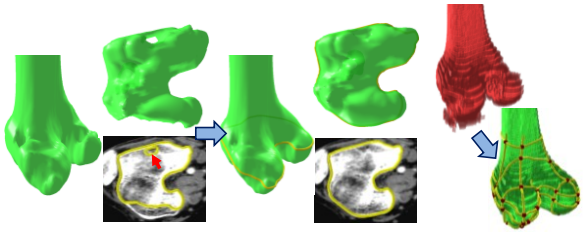

Figure 10. *Effect of contour deletion(left) and refined knee joint region(right). If the user clicks on an undesired contour, the system removes the hole in the boundary surface.*

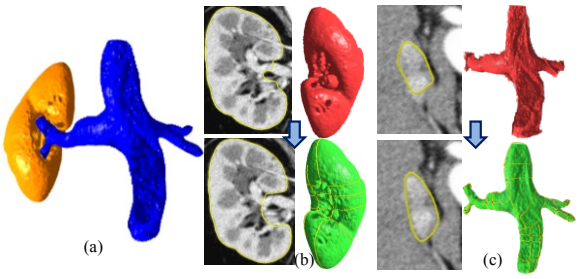

Figure 11. *Kidney and associated veins. The user replaced contours on complex and low contrast cross sections.*

Figure 11 shows a segmented kidney and associated veins from a CT image. The hilum of the kidney has a complicated blood vessel structure. The user must place a boundary between the kidney and the external blood vessel regions accurately (Figs. 11(b)). The vein in the CT image has an extremely low-contrast boundary, which is almost impossible to segment automatically. Our system allows the user to trace plausible contours intuitively (Figs. 11(c)).

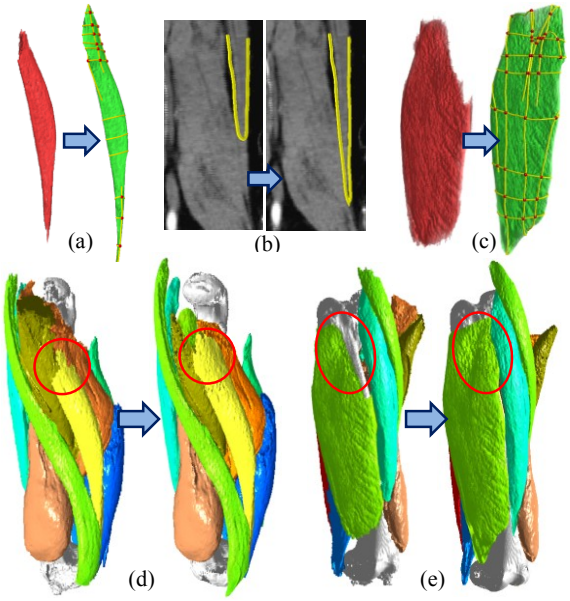

Figure 12. *The regions of 11 femoral muscles and 1 bone were refined. Two representative muscles are shown in (a-c). Very sharp regions are correctly reconstructed (red circles) by tracing narrow boundaries (b).*

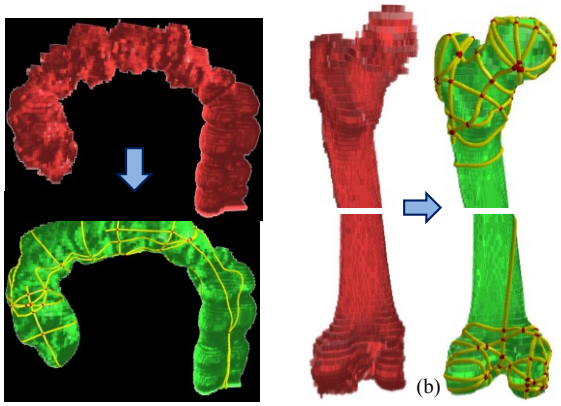

Figure 13. *Bowel and thigh bone refined by the test user.*

Muscles in CT images are also a difficult target for automatic algorithms because they usually have a low-contrast boundary. Refining 1 bone and 11 muscles with our system took less than 4 hours (Fig. 12). In this example, very sharp features of muscles were reconstructed (Figs. 12(a) and (b)). We believe that the user's input is essential for accurately tracing the ambiguous boundaries shown in these examples.

Finally, we performed a preliminary user study to verify the usability of our system. We asked a technician working on the development of medical imaging devices to refine segmentation data of a partial bowel and a thigh bone with our system. Although the subject has used 2D segmentation systems, he has never experienced 3D systems. After one hour tutorial, he successfully refined segmentation data as shown in Figure 13. It took about 45 minutes to reconstruct a part of bowel by placing about 20 contours (Fig. 13(b)). The user first refined contours along the curved axis of the bowel and later refined contours perpendicular to the axis. It also took about 15 minutes for refining each side of thigh bone (Fig. 13(a)). During modification, the drawing, point-dragging, and smooth-rubbing tools were evenly well used, but the snake-rubbing tool was almost useless since regions which requires modifications usually has ambiguous edges.

As we mentioned in Session 2, SketchSurface [AM07] and VolumeViewer [SLJ*09] applied contour-based mesh construction methods to volume segmentation purpose. In these systems, the segmentation process starts from scratch: the user draws multiple contours in the 3D space only with the cross-section visualisation. On the other hand, our system has an initial segmentation obtained with arbitrary algorithms. We believe that minor modification of contour shapes is much easier than drawing contours from scratch. In addition, while these systems are limited to closed and planar contours, we support to handle open and non-planar contours, which permit efficient local modification (e.g., sides of thigh bone in Fig. 10) and effective manipulation of curved objects (e.g., bowel in Fig. 13 (a)). Besides, it is not straightforward to apply these two systems [AM07, SLJ*09] to refining already extracted regions. These systems compute a boundary surface satisfying contour constraints without considering a detailed shape of the initial segmented region, and then they fail to maintain the important detail features, similarly to Figure 9 (e).

7. Conclusion

We designed a contour-based interface for efficiently refining the foreground region of medical images. We have provided a set of tools for placing, modifying, and deleting contour constraints for an intuitive refinement process. We also introduced a constraint-based surface-deformation algorithm that seamlessly combines the detail-preserving and LM-diffusing methods. Automatic systems have difficulty generating satisfactory segmentation results. Our system can serve as an efficient post-processing method for any existing automatic system; the user simply refines the region until completely satisfied.

Depending on the purpose of the segmentation, e.g., construction of a mesh model for simulation, our boundary surface is available as is, and our system allows refining the surface in more detail than the voxel level. Therefore, our system would be useful for constructing a model that currently available imaging devices cannot capture in sufficient resolution, e.g., microscopic images of cells.

One obvious limitation of our system is that it does not support arbitrary topological modifications. Providing a more flexible topology editing tool that supports not only deleting but also merging contours on the cross sectional surface is remaining as our future work. Another limitation is that it is difficult to efficiently refine thin strained or membranous regions, such as blood vessels or aponeurosis. In such regions, their medial axis shapes better explain the desired regions than the boundary shapes and then our contour-based interface often falls into tedious process. We would like to provide an axis based refinement tool special-

ized to such thin regions in the future. Another interesting future direction will be to combine our system with existing semi-automated systems more seamlessly. Our current system provides a separate post-refinement process. It would be very useful if the user could seamlessly move back and forth between coarse segmentation with existing automatic systems and fine-tuning with our system.

Acknowledgements

We thank Dr. Shin Yoshizawa for helpful comments and suggestions. The CT data shown in Figure 11 was provided by Prof. Kazuhide Makiyama and Prof. Yoshinobu Kubota in Yokohama City University. This paper was supported in part by Grant-in-Aid for Young Scientists (B) (22700115).

References

[AM07] ALIROTEH, M., MCINERNEY, T.: SketchSurfaces: Sketch-Line initialized deformable surfaces for efficient and controllable interactive 3D medical image segmentation, *ISVC* 2007, 542–553, 2007.

[BS08] BOTCH, M., SORKINE, O.: On linear variational surface deformation methods. IEEE TVCG, 14, 1(2008), 213–230.

[BVZ01] BOYKOV, Y., VEKSLER, O., ZABIH, R.: Fast approximate energy minimization via graph cuts. *IEEE Trans. PAMI, 23, 11*, 1222–1239, 2001.

[HPE*08] HE, L., PENG, Z., EVERDING, B., WANG, X., HAN, C. Y., WEISS, K. L., WEE, W. G.: A comparative study of deformable contour methods on medical image segmentation. *Image and Vision Computing, 26(2)*, 141–163.

[IMH05] IGARASHI, T., MOSCOVICH, T., HUGES, J. F.: As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24, 3*(2005), 1134–1141.

[KWT88] KASS, M., WITKIN, A., TERZOPOULOS, D.: Snakes: Active contour models, *Int. J. Comput. Vis., 1*, 321–331, 1988.

[LSTS04] LI, Y., SUN, J., TANG, C. K., SHUM, H. Y.: Lazysnapping. In *Proc. of ACM SIGGRAPH '04*, 303–308, 2004.

[LBD*08] LIU, L., BAJAJ, C., DEASY, J., LOW, D., JU, T.: Surface reconstruction from non-parallel curve networks. *Computer Graphics Forum, 27, 2*, 155–164, 2008.

[LC87] LORENSEN, W. E., CLINE, H. E.: Marching Cubes: A high resolution 3D surface construction algorithm, *Computer Graphics, 21(3)*, 163–169, 1987.

[NISA07] NEALEN, A., IGARASHI, T., SORKINE, O., ALEXA, M.: Fibermesh: Designing freeform surfaces with 3D curves. *ACM Trans. Graph. 26, 3* (2007).

[NSAC05] NEALEN, A., SORKINE, O., ALEXA, M., COHEN-OR, D.: A sketch-based interface for detail-preserving mesh editing. *ACM Trans. Graph. 24, 3*, 1142–1147, 2005.

[ONT05] OWADA, S., NIELSEN, F., IGARASHI, T.: Volume Catcher, In *Proc. of I3D '05*, 111–116, 2005.

[PXP98] PHAM, D. L., XU, C., PRINCE, J. L.: A survey of current methods in medical image segmentation. In *Technical Report JHU/ECE 99-01, The Johns Hopkins University,* 1998.

[SHN03] SHERBONDY, A., HOUSTON, M., NAPEL, S.: Fast volume segmentation with simultaneous visualization using programmable graphics hardware. In *Proc. of IEEE Vis.*, 171–176, 2003.

[Sor06] SORKINE, O.: Differential representations for mesh processing. *Computer Graphics Forum, 25,* 4(2006), 789–807.

[SLC*04] SORKINE, O., LIPMAN, Y., COHEN-OR, D., ALEXA, M., ROSSL, C., SEIDEL, H. P.: Laplacian surface editing. In *Proc. of SGP '04*, 179–188, 2004.

[SLJ*09] SOWELL, R., LIU, L., JU, T., GRIMM, C., ABRAHAM, C., GOKHROO, G., LOW, D.: VolumeViewer: an interactive tool for fitting surfaces to volume data. In *Proc. of SBIM '09*, 141–148, 2009.

[TLM03] TZENG, F. Y., LUM, E. B., MA, K.-L.: A novel interface for higher-dimensional classification of volume data. In *Proc. of IEEE Vis.*, 505–512, 2003.

[WBC*05] WANG, J., BHAT, P., COLBURN, R. A., AGRAWALA, M., COHEN, M.F.: Interactive video cutout. *ACM Trans. Graph., 24(3)*, 585–594, 2005.

[YZNC05] YUAN, X., ZHANG, N., NGUYEN, M. X., CHEN, B.: Volume cutout, *The Visual Computer*, *21(8-10)*, 745–754, 2005.