EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Bilateral Hermite Radial Basis Functions for Contour-based Volume Segmentation

Takashi Ijiri[1], Shin Yoshizawa[1], Yu Sato[2], Masaaki Ito[2], and Hideo Yokota[1]

[1]RIKEN, [2]National Cancer Center Hospital East

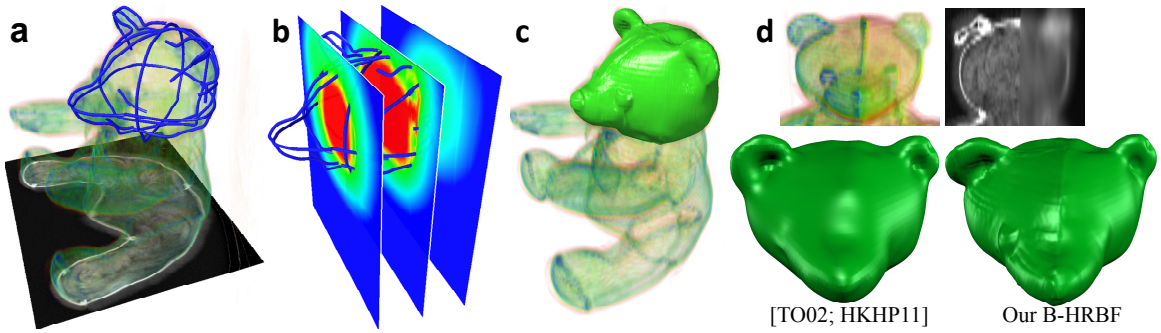

Figure 1: *Our segmentation example. We cut out the head region from the CT volume of a stuffed bear. From specified contours (a), we computed a scalar field (b) and generated a boundary from the field (c). We smoothed the right half of the volume and computed the boundaries (d) using a previous method (left) and our B-HRBF (right) with the same contours.*

Abstract

In this paper, we propose a novel contour-based volume image segmentation technique. Our technique is based on an implicit surface reconstruction strategy, whereby a signed scalar field is generated from user-specified contours. The key idea is to compute the scalar field in a joint spatial-range domain (i.e., bilateral domain) and resample its values on an image manifold. We introduce a new formulation of Hermite radial basis function (HRBF) interpolation to obtain the scalar field in the bilateral domain. In contrast to previous implicit methods, bilateral HRBF (B-HRBF) generates a segmentation boundary that passes through all contours, fits high-contrast image edges if they exist, and has a smooth shape in blurred areas of images. We also propose an acceleration scheme for computing B-HRBF to support a real-time and intuitive segmentation interface. In our experiments, we achieved high-quality segmentation results for regions of interest with high-contrast edges and blurred boundaries.

Categories and Subject Descriptors (according to ACM CCS): I.4.6 [Computer Graphics]: Segmentation, I.3.6 [Computer Graphics]: Interaction techniques, I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, Curve, surface, solid, and object representations.

1. Introduction

To obtain important information from volume images and construct mesh models for simulations, it is crucial to segment the volume. Due to the wide variety of imaging modalities and segmentation targets, fully automatic segmentation is very difficult and user interaction is still required.

Many semi-automatic segmentation methods have been published in recent decades. They can be roughly divided into seed-based and contour-based methods, in terms of the user-interface characteristics. Seed-based methods [AB94; BVZ01] allow users to extract a region of interest (ROI) by roughly specifying a small number of seeds. Such methods work very well for ROIs with clear image boundaries; however, they often generate errors around blurred or complex image areas, because the seeds provide only rough inside/outside information. A contour-based approach is a promising way to accurately excise ROIs with ambiguous boundaries [TO02; LBD*08; IY10]. Although contouring requires more manipulation than seeding, users can directly specify contours on ambiguous areas of images, and the system is able to optimize segmentation boundaries with much richer constraints. Many ROIs in biomedical images have ambiguous boundaries that only experts can detect. Segmenting such ROIs requires contour-based approaches.

Our goal is to develop a contour-based volume segmentation technique specialized for extracting ROIs with ambiguous image boundaries. Three requirements must be met to generate segmentation boundaries from contours: the boundary should i) pass through all contours; ii) fit high-contrast image edges, if they exist near the boundary; and iii) be smooth around blurred areas of images. Existing contour-based segmentation methods [TO02; LBD*08; IY10] focus only on the smoothness of the boundary (see Fig. 1d, left), and cannot fit the boundary to image edges.

This paper proposes a novel contour-based segmentation technique based on an implicit surface-reconstruction strategy. The key idea is to construct a scalar field from contours in a *joint spatial-range domain* (i.e., *bilateral domain*) and resample its values on an image manifold [SKM98]. While conventional implicit methods define segmentation boundaries as zero-level sets, our segmentation boundary is defined by the intersection between the image manifold and a zero-level set of the scalar field. Because the scalar field is generated by a Hermite-type interpolation with a radial basis function (RBF) on the image manifold, we refer to our new implicit function as the *bilateral Hermite radial basis function* (*B-HRBF*). Without explicitly calculating the intersection, our B-HRBF successfully generates the segmentation boundary that satisfies the three requirements summarized above. We also developed a new acceleration scheme for computing the B-HRBF.

To verify the feasibility of our technique, several real-world volumes were examined. Figure 1 illustrates our segmentation results for a computed tomography (CT) image of a stuffed animal with ambiguous boundaries. The resulting boundary fits to high-contrast edges well, and is smooth around blurred areas (Fig. 1d, right).

The contributions and benefits of this paper are below;

- A novel B-HRBF formulation for segmenting ROIs with high-contrast edges and ambiguous boundaries.
- A contour-based user interface for intuitive volume segmentation.
- An acceleration scheme for B-HRBF computation for supporting real-time segmentation.

2. Related Work

Our technique is related to previous methods such as surface reconstruction and image segmentation.

Surface reconstruction. Methods for reconstructing a surface from scattered data points (i.e., positions with normals) have been widely applied in various fields. There are two main surface-reconstruction methods: direct meshing and implicit approaches.

Direct meshing reconstructs a surface by directly connecting the input data points. Edelsbrunner and Nucke [EM94] defined the alpha shape as a generalized convex hull for scattered points, and presented an algorithm to reconstruct it based on Delaunay triangulation. Bernardini et al. [BMR*99] presented the ball-pivoting algorithm, which incrementally constructs triangles by rolling a ball on the scattered points. Amenta et al. [ABK98] applied Delaunay triangulation to input points, and subsequently trimmed unnecessary simplices. However, direct meshing assumes sufficiently dense point distributions. Hence, it is difficult to apply when data points exist only on contours and are non-uniformly distributed.

In implicit approaches, a signed scalar field, $f(\cdot)$, with a zero value at all scattered points $\mathbf{p}$ (i.e., $f(\mathbf{p}) = 0$) and positive/negative values inside/outside, is first computed; then its zero-level set is extracted. RBF interpolation is commonly used for computing the scalar field, because it can generate a smooth field from non-uniformly distributed data points [Duc77; CBC*01; TO02; KHR02; OBS04; BK05; Wen05]. However, it requires the definition of hand-tuned offset points (Fig. 2b). For instance, in a previous study [CBC*01], the authors generated two additional offset points at $\mathbf{p} \pm \varepsilon\mathbf{n}$ with associated values $\pm\varepsilon$, for each point $\mathbf{p}$ with a unit normal $\mathbf{n}$. In general, a single optimal choice for ε does not exist. To address this issue, some researchers have presented Hermite radial basis functions (HRBFs) that directly incorporate normals and gradients as $\nabla f(\mathbf{p}) = \mathbf{n}$ (Fig. 2c) [WSC06; BMS*10; MGV10].

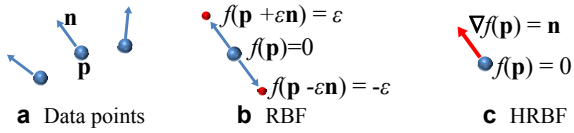

Figure 2: *Directional constraints of RBF and HRBF.*

Image Segmentation. Image segmentation is an active field of research, and many different segmentation approaches have been presented, such as thresholding, region growing, k-means clustering, mean-shift, active contours, graph cut, and so on [AB94; PXP98; Wir07]. We review methods that are closely related to our work. Active contour image segmentation was originally presented by Kass et al. [KWT88], and many extensions have been developed [HPE*08]. This method iteratively updates an evolving boundary (i.e., a curve in a two-dimensional (2D) domain and a surface in a three-dimensional (3D) domain) so as to minimize two types of energy: internal energy sensitive to the boundary shape and external energy sensitive to local image features around the boundary. Its level set formulations with RBFs [GBFP07; XM11] are able to handle topological changes of the evolving boundary. Although these methods can balance smoothing and image-edge-fitting effects, they are difficult to adapt to interactive segmentation with expert knowledge. Boykov et al. [BVZ01] and Li et al. [LSTS04] used a graph cut algorithm for image segmentation. This method formulates image segmentation as an energy minimization problem, and solves it by constructing an undirected weighted graph over the image pixels and computing its minimum cut. Boykov and Kolmogorov [BK03] extended this method to find minimal surfaces. However, it is difficult to apply this method in our context, because it does not control the boundary smoothness.

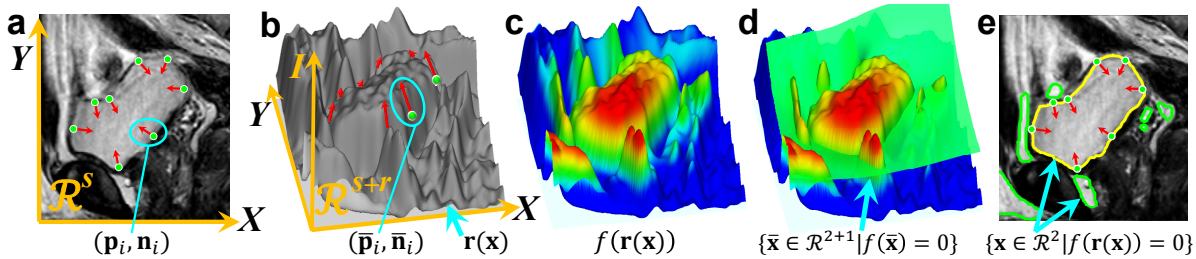

Figure 3: *An example of B-HRBF-based segmentation (s=2 and r=1). We specify eight data points on a 2D grayscale medical image (a). The image manifold* $\mathbf{r}(\mathbf{x})$ *forms a height function in a 3D joint domain* $\mathcal{R}^{2+1}$ *and the data points in* $\mathcal{R}^2$ *are mapped onto* $\mathbf{r}(\mathbf{x})$ *(b). We compute the B-HRBF* $f(\bar{\mathbf{x}})$*, evaluate its values on* $\mathbf{r}(\mathbf{x})$ *(c), and extract regions with positive field values (e). The segmentation boundary is equivalent to the intersection between* $\mathbf{r}(\mathbf{x})$ *and the zero-level set of* $f(\bar{\mathbf{x}})$ *(d).*

Contour-based volume segmentation. Recently, some researchers have presented contour-based 3D segmentation systems. Because they are based on the surface-reconstruction methods discussed above, they can be divided into direct meshing and implicit approaches. The method of Bruin et al. [BDP*05] reconstructs boundary surfaces from well-organizes contours by directly connecting contour vertices. The SketchSurface system [AM07] initializes boundary surfaces by quick-hull [BDH96], and subsequently applies active contour iterations. Liu et al. [LBD*08] initialize boundary surfaces using a Voronoi diagram-based algorithm, and then smooth them. All of these methods use direct meshing. Due to their surface-reconstruction algorithms, they require well-organized contours, and it is difficult to manage open or non-planar contours. Some other researchers have suggested implicit methods [TO02; HKHP11], constructing a signed scalar field from contours via RBF interpolation and then extracting its zero-level set. These methods can generate a smooth boundary surface from various types of contours, such as closed/open or planar/non-planar contours. However, it is difficult to fit a boundary surface to image edges.

3. B-HRBF for Image Segmentation

In this section, we introduce our B-HRBF formulation, and in the next section we present a contour-based interface that provides the conditions for constructing B-HRBF.

Given an *s*-dimensional *r*-channel image, we consider an (*s*+*r*)-dimensional *joint domain* $\mathcal{R}^{s+r}$ that connects an *s*-dimensional *spatial domain* $\mathcal{R}^s$ and an *r*-dimensional *range domain* $\mathcal{R}^r$. Grid pixels are arranged in the spatial domain, and pixel values (i.e., color) are represented in the range domain. In these domains, Euclidean metrics are assumed. Connecting a spatial position $\mathbf{x} \in \mathcal{R}^s$ and its associated range value $\mathbf{I}(\mathbf{x}) \in \mathcal{R}^r$, an *s*-manifold embedded in $\mathcal{R}^{s+r}$ (i.e., the *image manifold*) is defined by

$$\mathbf{r}(\mathbf{x}) = \begin{pmatrix} \mathbf{x} \\ w^c \mathbf{I}(\mathbf{x}) \end{pmatrix} \quad \ldots (1)$$

where $\mathbf{r}(\mathbf{x}) \in \mathcal{R}^{s+r}$ and w^c is a scaling coefficient to control the influence of image features on segmentation. A similar idea has been employed to accelerate edge-aware image filtering [PD06].

From user-specified multiple contours, we compute a smooth scalar field $f(\bar{\mathbf{x}}) \in \mathcal{R}$ in the joint domain, where $\bar{\mathbf{x}} \in \mathcal{R}^{s+r}$ is an arbitrary point in the joint domain. Unlike in standard implicit surface reconstruction methods, our segmentation boundary is given by the intersection between the image manifold $\mathbf{r}(\mathbf{x})$ and the zero-level set of the scalar field $\{\bar{\mathbf{x}} \in \mathcal{R}^{s+r} | f(\bar{\mathbf{x}}) = 0\}$. This intersection is equivalent to the zero-level set of the scalar field on the image manifold $\{\mathbf{x} \in \mathcal{R}^s | f(\mathbf{r}(\mathbf{x})) = 0\}$. Therefore, we evaluate the scalar field only on the image manifold and extract regions with positive field values as the foreground $\{\mathbf{x} \in \mathcal{R}^s | f(\mathbf{r}(\mathbf{x})) \geq 0\}$.

3.1 Bilateral Hermite Radial Basis Function

Assume that multiple contours are specified in the spatial domain, and *N* data points are generated by resampling the contours as $\{(\mathbf{p}_i, \mathbf{n}_i) | \mathbf{p}_i, \mathbf{n}_i \in \mathcal{R}^s, i = 1, \ldots, N\}$. Each data point consists of a point $\mathbf{p}_i$ through which a boundary passes and a unit normal vector of the boundary $\mathbf{n}_i$ at $\mathbf{p}_i$. Each data point has the associated range value $\mathbf{I}(\mathbf{p}_i) \in \mathcal{R}^r$. We map each point $\mathbf{p}_i$ and normal $\mathbf{n}_i$ into the joint domain by

$$\bar{\mathbf{p}}_i = \mathbf{r}(\mathbf{p}_i), \qquad \bar{\mathbf{n}}_i = \mathbf{J}\,\mathbf{n}_i / ||\mathbf{J}\,\mathbf{n}_i|| \quad \ldots (2)$$

where $\bar{\mathbf{p}}_i, \bar{\mathbf{n}}_i \in \mathcal{R}^{s+r}$ and $\mathbf{J} \in \mathcal{R}^{(s+r)\times s}$ is the Jacobian matrix of $\mathbf{r}(\mathbf{x})$. Note that each point is mapped to a corresponding position on the image manifold, and each normal is mapped so that it is on the tangent plane of the image manifold (Fig. 3b).

Our *Bilateral Hermite Radial Basis Function* (*B-HRBF*) is given by

$$f(\bar{\mathbf{x}}) = \sum_{i=1}^{N} \left(\alpha_i \varphi(\|\bar{\mathbf{x}} - \bar{\mathbf{p}}_i\|) - \boldsymbol{\beta}_i \cdot \nabla\varphi(\|\bar{\mathbf{x}} - \bar{\mathbf{p}}_i\|) \right) + P(\bar{\mathbf{x}}) \quad \ldots (3)$$

where $P(\bar{\mathbf{x}})$ is a polynomial term and $\alpha_i \in \mathcal{R}$ and $\boldsymbol{\beta}_i \in \mathcal{R}^{s+r}$ are unknown coefficients. The coefficients α_i and $\boldsymbol{\beta}_i$ are determined by solving the interpolation and gradient constraints, as follows:

$$f(\bar{\mathbf{p}}_i) = 0, \quad \nabla f(\bar{\mathbf{p}}_i) = w^b \bar{\mathbf{n}}_i + (1 - w^b) \begin{pmatrix} \mathbf{n}_i \\ \mathbf{0} \end{pmatrix} \quad \ldots (4)$$

where $w^b \in [0,1]$ is a parameter. The linear combination in the gradient constraint in Eq. (4) is employed to avoid false positive regions (see Section 3.2). Our experiments suggest that the commonly used tri-harmonic kernel $\varphi(t) = t^3$ with a linear polynomial $P(\bar{\mathbf{x}}) = \mathbf{a} \cdot \bar{\mathbf{x}} + b$ provides satis-

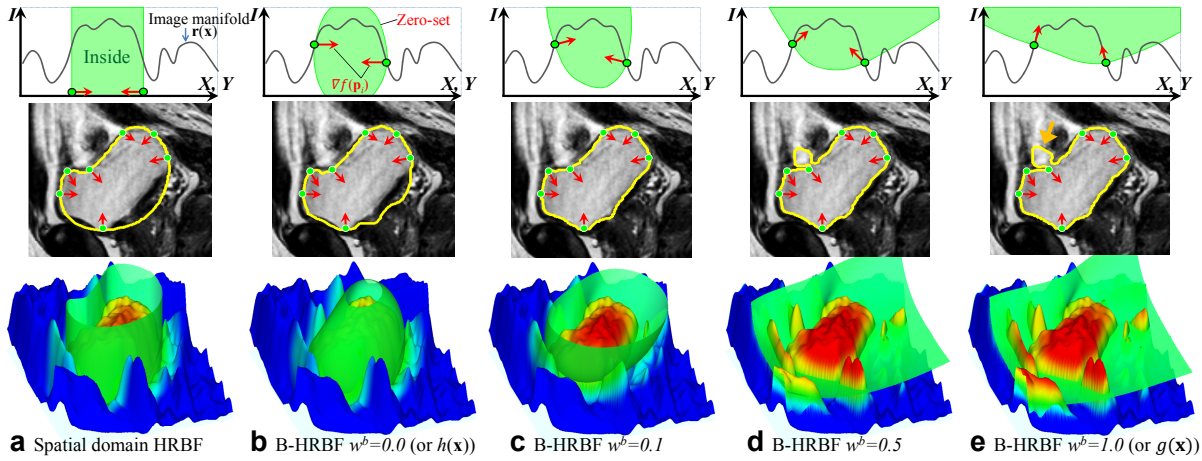

Figure 5: *Analysis of the gradient constraint in Eq. (4). The top row illustrates the constraints and the zero-level set of the scalar field in 2D. The middle and bottom rows show the segmentation boundary and the zero-level sets in the joint domain, respectively. The spatial domain HRBF $H(\mathbf{x})$ does not depend on the range domain (a) whereas the extended spatial domain HRBF $h(\bar{\mathbf{x}})$ does (b). (b–e) Results generated using our B-HRBF in Eqs. (3) and (4) with different w^b.*

factory results in terms of shape aesthetics. The unit normal $\mathbf{n}_i$ in Eqs. (2) and (4) is computed from the specified contours and their associated image values (see Section 4.1).

Figure 3 shows an example of 2D grayscale image segmentation (i.e., s = 2 and r = 1). Our segmentation technique successfully meets the three requirements. The segmentation boundary passes through all data points. High-contrast image edges form steep slopes on the image manifold. In such areas, the intersection between $\mathbf{r}(\mathbf{x})$ and $\{\bar{\mathbf{x}} \in \mathcal{R}^{s+r} | f(\bar{\mathbf{x}}) = 0\}$ tends to pass through the slopes, and the segmentation boundary fits the image edges. Blurred image areas form comparably flat shapes on the image manifold. In such areas, the smoothness of the B-HRBF is dominant comparing to the edge fitting effect, and the segmentation boundary becomes smoother.

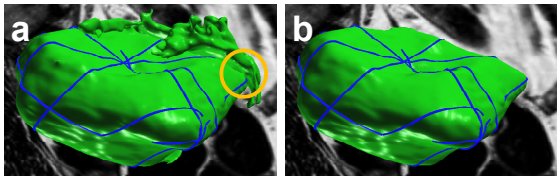

Figure 4: *3D false positive regions generated by $g(\bar{\mathbf{x}})$ (a)* are avoided by $f(\bar{\mathbf{x}})$ (*b*). See also Fig. 5e for a 2D example.

3.2 Analysis of the Gradient Constraint in B-HRBF

Let $g(\bar{\mathbf{x}}) \in \mathcal{R}$ be a B-HRBF in Eq. (3) constructed by solving the constraints (i.e., Eq. (4) with w^b=1) as follows:

$$g(\bar{\mathbf{p}}_i) = 0, \ \nabla g(\bar{\mathbf{p}}_i) = \bar{\mathbf{n}}_i.$$

Such straightforward formulations have often led to *false positive regions*, in which the ROI and undesired positive regions are connected by "narrow tunnels" (Fig. 4a and the orange arrow in Fig. 5e). To avoid such errors, we use the linear combination in the gradient constraint in Eq. (4).

The idea behind the linear combination in Eq. (4) is to blend the two scalar fields generated in the joint and spatial domains. A key observation is that false positive regions are not an issue if we compute a scalar field in the spatial domain and evaluate it at pixel positions [TO02; HKHP11]. Let us denote a standard HRBF [MGV10] in the spatial domain by $H(\mathbf{x}) \in \mathcal{R}$ with $H(\mathbf{p}_i) = 0$ and $\nabla H(\mathbf{p}_i) = \mathbf{n}_i$. Figure 5a shows the segmentation result obtained with $H(\mathbf{x})$. Although a false positive region does not appear, the boundary does not trace the image edges. Consider an extension of the spatial domain HRBF $H(\mathbf{x})$ to the joint domain such that the extended function $h(\bar{\mathbf{x}}) \in \mathcal{R}$ is given by the B-HRBF in Eq. (3) with the following constraints (i.e., Eq. (4) with w^b=0):

$$h(\bar{\mathbf{p}}_i) = 0, \ \nabla h(\bar{\mathbf{p}}_i) = \begin{pmatrix} \mathbf{n}_i \\ \mathbf{0} \end{pmatrix}.$$

While we move each point $\mathbf{p}_i$ onto the image manifold $\bar{\mathbf{p}}_i$ using Eq. (2), we fix the normal direction in the spatial domain as $(\mathbf{n}_i \ \mathbf{0})^T$. Figure 5b shows the segmentation result with this extended spatial domain HRBF $h(\bar{\mathbf{x}})$.

Combining the joint domain HRBF $g(\bar{\mathbf{x}})$ and the extended spatial domain HRBF $h(\bar{\mathbf{x}})$ with a coefficient w^b as

$$f(\bar{\mathbf{x}}) = w^b g(\bar{\mathbf{x}}) + (1 - w^b) h(\bar{\mathbf{x}})$$

results in our B-HRBF in Eqs. (3) and (4). We do not need to compute $g(\bar{\mathbf{x}})$ and $h(\bar{\mathbf{x}})$ separately.

The linear combination in the gradient constraint of Eq. (4) simply scales the range domain elements of $\bar{\mathbf{n}}_i$ by w^b. We illustrate this scaling effect in Figure 5b–e. With a small w^b, $\nabla f(\bar{\mathbf{p}}_i)$ is oriented more orthogonally to a tangent plane of the image manifold at $\bar{\mathbf{p}}_i$, and the zero-level set of $f(\bar{\mathbf{x}})$ and the image manifold do not readily intersect. As a result, undesired positive regions are inhibited. With a large w^b, $\nabla f(\bar{\mathbf{p}}_i)$ is more parallel to the tangent plane at $\bar{\mathbf{p}}_i$. As a result, the zero-level set of $f(\bar{\mathbf{x}})$ tends to cross steep slopes on the image manifold, and the boundary fits the image edges. Proper selection of w^b successfully avoids false positive regions, while maintaining the edge-fitting effect (Figs. 4b and 5c). In our experience, the fixed value w^b = 0.1 works well.

3.3 Evaluating the B-HRBF on the Image Manifold

When extracting the foreground, we evaluate the B-HRBF on the image manifold; i.e., $f(\mathbf{r}(\mathbf{x}))$. However, evaluating the B-HRBF at all pixels of the image manifold is time-consuming, and may generate *extra positive regions* which are undesired positive regions disconnected from the ROI (green curves in Fig. 3e). We avoid such problems by employing the surface-tracking marching cubes algorithm [SFYC*96]. This algorithm starts from several seed cells, and visits only those cells that are likely to compose parts of a connected surface. The cell represents a unit block to resample a scalar field and generate surface patches. In our setup, seed cells are determined by evaluating the B-HRBF at neighboring pixels of each data point $\mathbf{p}_i$ (e.g., 9 neighbors for 2D and 27 neighbors for 3D). Because this algorithm evaluates the B-HRBF only at pixels around the boundary of an ROI, it is usually much faster than the all-pixel evaluation, and it also avoids detecting extra positive regions that are not connected to the ROI. In Figure 3(c-e), several peak-shaped regions around the ROI have positive values. Surface-tracking extracts only the boundary of the ROI (yellow curve in Fig. 3e).

4. Contour-based Volume Segmentation

This section presents our contour-based interface for volume segmentation. Until a desired segmentation boundary is obtained, the user repeats two manipulations: cross-section placement and contour specification. For the former, we provide a cut-stroke interface [IY10] to generate curved cross-sections along a stroke (Fig. 6a), as well as three *x*–*y*, *y*–*z*, and *z*–*x* planes. After placing a cross-section at a position where the ROI appears well, the user specifies a contour by sequentially clicking points on the cross-section. We place control points at the clicked positions and interpolate them with a degree 3 B-spline curve (Fig. 6b). The user can select an open or closed contour and switch the ROI sides (red marks in Fig. 6b). When finishing a contour specification, the user *stores* it and moves to the next contour. The user can also *activate* an already stored contour and re-edit it by adding, moving, or deleting control points. After each manipulation, the segmentation boundary is immediately updated. We compute the segmentation using a background thread, and thus the user is able to edit contours without waiting.

4.1 Modeling Normals from Contours

We generate a set of data points, $\mathbf{p}_i \in \mathcal{R}^s$, by resampling all contours at an interval of w^{int}. Let $\mathbf{t}_i$ be a tangent vector of the contour at $\mathbf{p}_i$ and $\mathbf{c}_i$ be a normal vector of the cross-section surface on which the contour is specified. In [HKHP11], the normal vector of the boundary surface $\mathbf{n}_i^0 \in \mathcal{R}^s$ at $\mathbf{p}_i$ was computed as

$$\mathbf{n}_i^0 = \pm(\mathbf{c}_i \times \mathbf{t}_i)/||\mathbf{c}_i \times \mathbf{t}_i||, \quad \ldots(5)$$

where the sign is determined from the ROI side (Fig. 7d). This method is limited to generating normals parallel to the cross-section (Fig. 7b), and works well only when the cross-section orthogonally intersects with the boundary of the ROI. However, cross-sections are usually slanted with respect to the boundary. To address this issue, we provide the following heuristic approach to automatically generate appropriate normals for Eqs. (2) and (4).

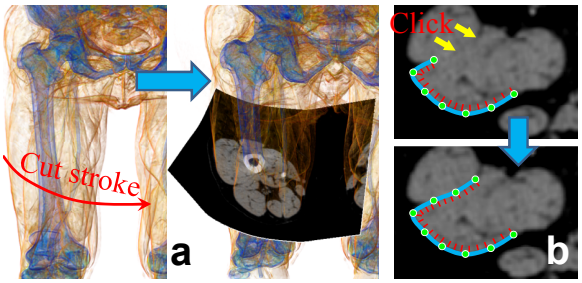

Figure 6: *User interface for specifying contours.*

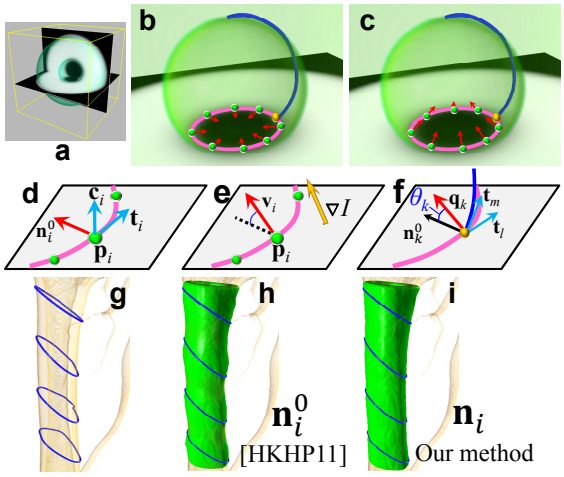

Figure 7: *Modeling normals. We specified two contours (b,c) on an artificial volume (a). While the previous method (d) was limited to planar normals (b), our two heuristic approaches (e,f) generate appropriate normals (c). We illustrate the effect of our normal modeling in (g-i).*

Because we want to fit a segmentation boundary to image edges, a normal vector $\mathbf{n}_i$ should be oriented along the image intensity gradient $\nabla I(\mathbf{p}_i)$ at $\mathbf{p}_i$. We consider the following vector:

$$\mathbf{v}_i = \begin{cases} \mathbf{n}_i^0 + w^n \nabla I(\mathbf{p}_i) & \text{if } \mathbf{n}_i^0 \cdot \nabla I(\mathbf{p}_i) > 0 \\ \mathbf{n}_i^0 - w^n \nabla I(\mathbf{p}_i) & \text{otherwise} \end{cases} \quad \ldots(6)$$

where w^n is a parameter (Fig. 7e). Because $\mathbf{t}_i$ and $\mathbf{n}_i$ should be orthogonal to each other, $\mathbf{v}_i$ is projected onto a plane whose normal is equal to $\mathbf{t}_i$ and is normalized. Using this $\mathbf{v}_i$ as the normal $\mathbf{n}_i$ in Eqs. (2) and (4) works well around image areas with high-contrast edges.

In contrast, in blurred areas without obvious edges, the user often specifies many contours, and thus there are many intersections among contours. In such areas, we first search all intersections between one contour and all others. At the *k*-th intersection, the intersection normal is obtained as $\mathbf{q}_k=(\mathbf{t}_l\times\mathbf{t}_m)/||\mathbf{t}_l\times\mathbf{t}_m||$, where $\mathbf{t}_l$ and $\mathbf{t}_m$ are tangent vectors of the two intersecting contours (Fig. 7f). We approximate the

desired normal $\mathbf{n}_i$ by blending the intersection normals $\mathbf{q}_k$ with respect to their angles to $\mathbf{n}_k^0$ of Eq. (5) so that the desired normal is appropriately influenced by the neighbor intersections. Finally, the normal vector $\mathbf{n}_i$ at $\mathbf{p}_i$, used in Eqs. (2) and (4), is given by switching the above two cases (edge and blurred areas) as follows;

$$\mathbf{n}_i = \begin{cases} \mathbf{v}_i & \text{if } d_i > w^d \\ \mathbf{R}(\mathbf{t}_i, \theta_i)\mathbf{n}_i^0 & otherwise \end{cases} \quad \dots (7)$$

where d_i is the distance from point $\mathbf{p}_i$ to its nearest intersection along the contour, w^d is a parameter, and $\mathbf{R}(\mathbf{t}_i,\theta_i)$ is a rotation matrix along $\mathbf{t}_i$ with an angle θ_i. We apply Gaussian smoothing by 10 iterations, using three adjacent contour vertices to blend the angles at the intersections, in order to compute θ_i at $\mathbf{p}_i$.

Figure 7g-i shows the effects of the normals we obtain in this way. In the example, four planar contours are specified on a femur shaft. They are intentionally slanted with respect to the ROI (Fig. 7g). Using $\mathbf{n}_k^0$ as in [HKHP11], the incorrect normal directions cause undesirable artifacts (Fig. 7h). In contrast, our normals in Eq. (7) successfully avoid such artifacts (Fig. 7i). There are three parameters: w^{int}, w^n, and w^d. Although we have to tune w^{int} depending on the target, fixed values such as $w^n = 0.5$ and $w^d = 4.0w^{int}$ work well in our experience.

5. Solving the Linear System of the B-HRBF

To obtain the $\alpha_i, \boldsymbol{\beta}_i, \mathbf{a}$, and b of B-HRBF in Eq. (3), we solve Eq. (4) with orthogonality conditions $\sum_{i=0}^{N} \alpha_i = 0$ and $\sum_{i=0}^{N} \alpha_i \bar{\mathbf{p}}_i + \boldsymbol{\beta}_i = \mathbf{0}$. This yields a $B(N+1)\times B(N+1)$ dense linear system, represented in block form as

$$\begin{pmatrix} \mathbf{0} & \mathbf{S}_1^T & \cdots & \mathbf{S}_N^T \\ \mathbf{S}_1 & \mathbf{K}_{1,1} & \cdots & \mathbf{K}_{1,N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{S}_N & \mathbf{K}_{N,1} & \cdots & \mathbf{K}_{N,N} \end{pmatrix} \begin{pmatrix} \mathbf{s} \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_N \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_N \end{pmatrix}, \quad \dots (8)$$

where $B=(1+s+r)$ is the size of a block. We refer to the coefficient matrix of this linear system (8) as the *B-HRBF matrix* $\mathbf{A}$. The blocks $\mathbf{K}_{i,j}$ and $\mathbf{S}_i \in \mathcal{R}^{B\times B}$, and vectors $\mathbf{s}$, $\mathbf{w}_i$, and $\mathbf{c}_i \in \mathcal{R}^B$ are defined as

$$\mathbf{K}_{i,j} = \begin{pmatrix} \varphi(\|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_j\|) & -\nabla\varphi(\|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_j\|)^T \\ \nabla\varphi(\|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_j\|) & -\mathrm{H}\varphi(\|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_j\|) \end{pmatrix},$$
$$\mathbf{S}_i = \begin{pmatrix} \bar{\mathbf{p}}_i^T & 1 \\ \mathbf{E} & \mathbf{0} \end{pmatrix}, \mathbf{s} = \begin{pmatrix} \mathbf{a} \\ \mathrm{b} \end{pmatrix}, \mathbf{w}_i = \begin{pmatrix} \alpha_i \\ \boldsymbol{\beta}_i \end{pmatrix}, \mathbf{c}_i = \begin{pmatrix} 0 \\ \bar{\mathbf{n}}_i \end{pmatrix}, \dots (9)$$

where $\mathbf{E} \in \mathcal{R}^{(s+r)\times(s+r)}$ is a unit matrix and $\mathrm{H}\varphi \in \mathcal{R}^{(s+r)\times(s+r)}$ is the Hessian matrix of the kernel φ. As long as the data points $\bar{\mathbf{p}}_i$ are pairwise distinct, unique solutions are determined from this system [BMS*10]. The blocks related to the polynomial term $\mathbf{S}_i$ are placed in the top and left of the B-HRBF matrix to efficiently recycle the decomposition. Although the B-HRBF matrix is symmetric, its diagonal entries are zero, and thus a fast LDL decomposition-based solver does not generate a stable solution [PTVF07]. Then we apply an LU decomposition-based method; we decompose the B-HRBF matrix in the LU form and perform backward/forward substitutions.

5.1 Recycling LU-Decomposition for Acceleration

To efficiently decompose the B-HRBF matrix into the LU form, we present a recycling scheme based on the Crout [PTVF07] and Gondzio [Gon92] algorithms.

The Crout algorithm is one of the best-known LU decomposition algorithms. It stores the lower and upper triangle matrices, **L** and **U**, in one matrix, in which all diagonal elements of **L** are fixed as 1 and omitted. The algorithm visits each element of the original matrix **A** by column, from left to right, and within each column, from top to bottom, and modifies the element of **A** into an element of **L** or **U** (Fig. 8a). When computing elements of **L** in each column, the algorithm performs row permutations so as to reduce rounding errors (i.e., pivoting).

The full-scratch LU decomposition by the Crout algorithm takes $O(N^3)$ amount of time for an $N\times N$ matrix, and is therefore time-consuming. In our technique, the user edits only one *activated* contour at a time, and the blocks of the B-HRBF matrix related to the *stored* contours do not vary during editing. Thus, it is possible to accelerate the LU decomposition by reusing blocks of LU-decomposed matrices related to the stored contours.

Suppose that N data points are generated from the stored contours and the B-HRBF matrix **A** in Eq. (8) has been decomposed as

$$\mathbf{PAQ} = \mathbf{LU} \quad \dots (10)$$

where **L** and **U** are lower and upper triangle matrices, and **P** and **Q** are the row and column permutation matrices induced by pivoting of the Crout and Gondzio algorithms.

Adding a new contour. Under these conditions, the user adds and edits an activated contour. If the activated contour generates M data points, a new $B(1+N+M)\times B(1+N+M)$ B-HRBF matrix is obtained as

$$\begin{pmatrix} \mathbf{A} & \mathbf{A}_{21}{}^T \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}$$

where $\mathbf{A}_{21} \in \mathcal{R}^{BM\times B(1+N)}$ and $\mathbf{A}_{22} \in \mathcal{R}^{BM\times BM}$ are blocks of the activated contour's data points generated using Eq. (9). Note that the number of new data points, M, may vary dynamically according to contour modification. By multiplying the permutation matrices, we obtain

$$\begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{E} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{A}_{21}{}^T \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{E} \end{pmatrix} = \begin{pmatrix} \mathbf{LU} & \mathbf{PA}_{21}{}^T \\ \mathbf{A}_{21}\mathbf{Q} & \mathbf{A}_{22} \end{pmatrix} \dots (11)$$

where $\mathbf{E} \in \mathcal{R}^{BM\times BM}$ is a unit matrix. This is equivalent to a halfway result in which the Crout algorithm finishes decomposing the top-left block. We restart the algorithm from this point to decompose the rest (Fig. 8b). This partial LU decomposition takes $O(N^2M+NM^2+M^3)$ amount of time, and is therefore much faster than the full factorization (i.e., $O((N+M)^3)$) when $M<<N$.

Activating a stored contour. Under the same conditions as those assumed in Eq. (10), the user can also select a stored contour to activate it. Suppose that the user selects a contour that corresponds to $\{k+1, \dots, k+W\}$ rows and columns of the B-HRBF matrix **A** (Fig. 8c). To apply the

recycling LU decomposition scheme shown in Figure 8b, we have to perform a row/column permutation so that all corresponding rows/columns move to the bottom/right of the matrix while maintaining the LU form (Fig. 8d). This can be done simply by computing the full-scratch LU decomposition of a permutated B-HRBF matrix; however, it is time-consuming. Instead, we adopt the Gondzio algorithm [Gon92]. Given LU-decomposed matrices, this algorithm flips successive rows (or columns) of the original matrix and updates the **L** and **U** matrices so that they maintain the LU feature (Fig. 8e,f). First, all corresponding rows are moved to the bottom one-by-one by iteratively applying the Gondzio flipping operation, and then all columns are similarly moved to the right.

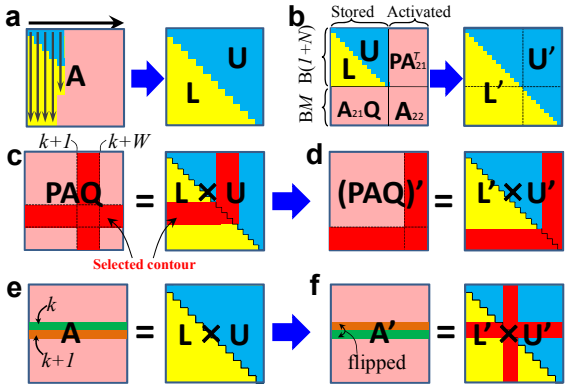

Figure 8: *Recycling LU decomposition. We run the Crout iterations (a) only in blocks related to the activated contour (pink blocks in (b)). When a stored contour is activated, we move the related rows/columns (highlighted in red) to the bottom/right of the matrix (c,d). The Gondzio algorithm updates the LU matrices after successive row flipping (e, f).*

The Gondzio flipping operation takes O(N–k) amount of time to flip the k-th and k+1-th rows of an N×N matrix. Then it takes O((N–k)^2) amount of time to move the k-th row to the bottom by applying the operation N–k times. Therefore, moving {k+1, …, k+W} rows takes O(W(N–k)^2) amount of time, much faster than the full-scratch decomposition of an N×N matrix (i.e., O(N^3) time) when W<<N or k is large enough. Unfortunately, the computational efficiency of this scheme is not obvious when M is large and k is small. We select the full-scratch decomposition when W>N/2 and k<N/2.

6. Results and Discussion

We examined our technique on several real-world volumes (Figs. 1, 9–12). All biomedical segmentation results were obtained in consultation with physicians and biologists. We also compared our technique to other commonly used segmentation methods [BVZ01; AB94] and state-of-the-art implicit approaches [TO02; MGV10; HKHP11] (Figs. 13 and 14).

Our method has two parameters that have to be tuned in accordance with the segmentation target, namely w^c, the scaling coefficient in Eq. (1); and w^{int}, the interval of contour resampling (Section 4.1). The tuning of w^c is intuitive, because it controls the influence of the image features. With small w^c, image features are given little consideration and a smooth boundary is obtained. With large w^c, image features become dominant, and the boundary fits the image edges. Fine tuning of w^{int} is unnecessary, because if w^{int} is small enough, it does not affect segmentation quality. However, it does influence computational time, since a small resampling interval generates many data points. We choose w^{int} so as to not generate too many data points, such as 6.0v (Fig. 1), 4.0v (Figs. 9–11), and 8.0v (Fig. 12), where v is the average of the voxel pitches on the x-, y-, and z-axes. We also apply a 5×5×5 kernel Gaussian filter to the input image three times to obtain visually pleasing results. Although other interpolation methods work fine with our technique, we apply the tri-linear interpolation to obtain a continuous **I**(**x**) from input images.

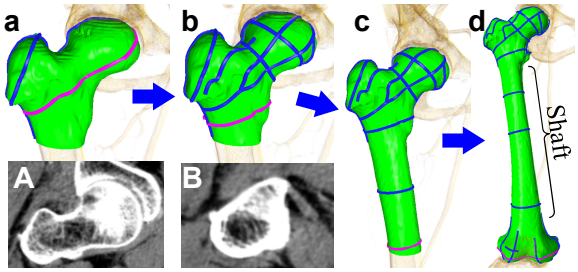

Figure 9: *Our segmentation process. Stored contours are highlighted in blue, and the activated contours are in red.*

Figure 9 illustrates the femur segmentation process using our technique. The user incrementally adds contours until a desired result is obtained. While previous contour-based methods have focused only on the smoothness of the boundary surface [LBD*08; HKHP11; BMS*10], our technique fits boundaries to high-contrast image edges. Therefore, fewer contours are necessary in our method than in previous methods. For example, since the shaft of the femur has high-contrast edges, three contours were enough to accurately extract the shaft shape (Fig. 9d).

For the example shown in Figure 10, we extracted all bones and muscles of the right thigh from a CT image. The joints of bones usually have complex boundaries (Fig. 9A), and seed-based methods often result in errors. In addition, boundaries between contacting muscles are low contrast and are difficult to detect automatically (Fig. 10A,B). Since our technique allows the user to directly specify contour constraints around ambiguous boundaries, such difficult ROIs are correctly segmented quickly. Each region in Figure 10 was excised in less than 30 min. We would like to emphasize that a ROI with an ambiguous image boundary requires slice-by-slice full manual segmentation, since most semi-automatic methods are not applicable (see Fig. 13). Our technique achieves large improvement in time comparing to the full manual segmentation.

In Figure 11, we extracted the anal sphincter and tumor regions from *magnetic resonance imaging* (MRI) images of the abdomens of two patients. Each anal sphincter region was segmented in less than 30 min by placing 10–15 con-

tours, and each tumor was segmented in 5 min, using about 4 contours. Medical specialists confirmed that these segmentation results are accurate enough to obtain significant information for scheduling and simulating surgeries. Because our technique makes it possible to extract these regions rapidly, such 3D shapes can be extracted before surgery. Applying our technique to clinical settings to enhance the quality of surgery is our ongoing goal.

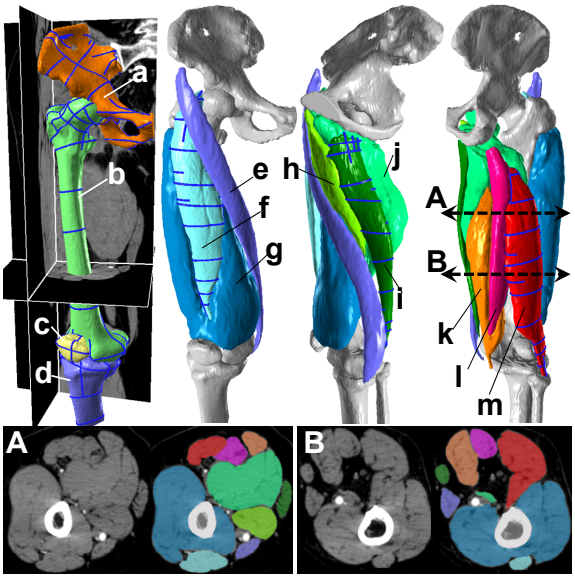

Figure 10: *Thigh segmentation. a: pelvis. b: femur. c: patella. d: tibia. e: satorius. f: rectus-femoris. g: vastus. h: adductor longus. i: gracilis. j: adductor-magnus. k:semimembranosus. l:semitendinosus. m:biceps-femoris.[Gra04] We set w^c=0.025 for bones and w^c=0.06 for muscles.*

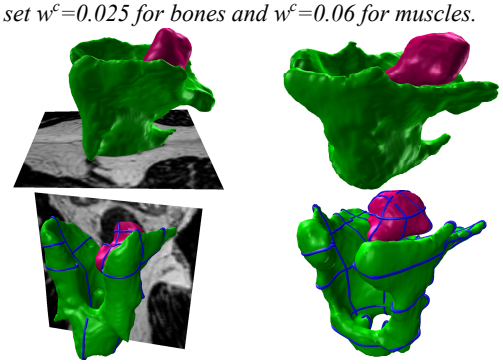

Figure 11: *Anal sphincter (green) and tumor (red) regions excised from MRI images. We set w^c = 0.15.*

Figure 12 shows an example of segmentation of a mouse image obtained using a microscopic 3D slicing device. We extracted multiple organs for anatomical analysis of genetically mutated and wild-type mice. For this color volume, we used the *CIE Lab* color domain because its Euclidian distance is known to be strongly correlated with perceptual color differences in humans [TM98]. We constructed an image manifold in $\mathcal{R}^{3+3}$ as $\mathbf{r}(\mathbf{x}) = (\mathbf{x}, w^c L(\mathbf{x}), w^c a(\mathbf{x}), w^c b(\mathbf{x}))^T$, where $L(\mathbf{x})$, $a(\mathbf{x})$, and $b(\mathbf{x})$ correspond to the L, a, and b color values at $\mathbf{x} \in \mathcal{R}^3$, respectively. Similar to other examples, the generated boundary correctly passed through high-contrast edges of the color volume. Note that we extracted multiple ROIs one-by-one, and overlapping regions of adjacent ROIs were selected manually. Computing multiple ROIs simultaneously remains as future work.

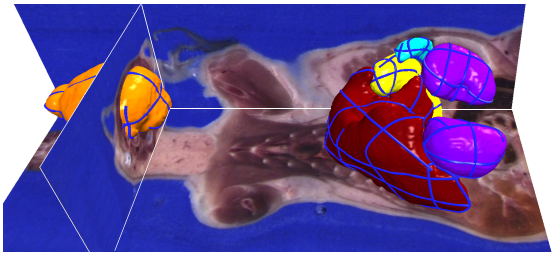

Figure 12: *Color volume segmentation. Brain (orange), kidneys (purple), liver (red), lien (light blue), and stomach (yellow) regions were extracted from a color volume of a mouse. We set w^c = 0.1.*

Figure 13 compares our method with common segmentation methods, such as graph cut [BVZ01] and region growing [AB94], using 2D CT images. When the target ROI has a clear image boundary (top row of Fig. 13), all methods achieve accurate segmentation. However, if the ROI has a blurred and ambiguous image boundary (bottom row in Fig. 13), the graph cut fails to generate a smooth boundary shape, and region growing fails to stop at the appropriate position. Most segmentation methods that rely on local image features often cause similar segmentation errors around ambiguous image areas. In contrast, our method achieves accurate segmentation, since it attempts to generate smooth boundary shapes around image areas without high contrast edges.

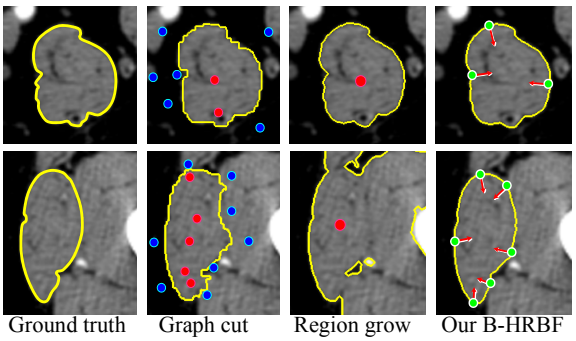

Figure.13. *Comparison with common segmentation methods. Ground truths were manually generated by an expert. In graph cut [BVZ01], we specified inside(blue)/outside (red) constraints. In region growing [AB94], we specified a seed (red). In B-HRBF, we specified 2D points with normals (red arrow). The yellow curves are the resulting segmentation boundaries.*

To compare our technique to previous implicit surface-reconstruction methods [TO02; MGV10; HKHP11], we computed segmentation boundaries of the femur region in Figure 9d, using the same set of contours by four different implicit functions (Fig. 14a-d). Around the top joint (i.e., femur head), where contours were densely specified, the four functions resulted in similar boundary surfaces. However, the functions with spatial domain representation

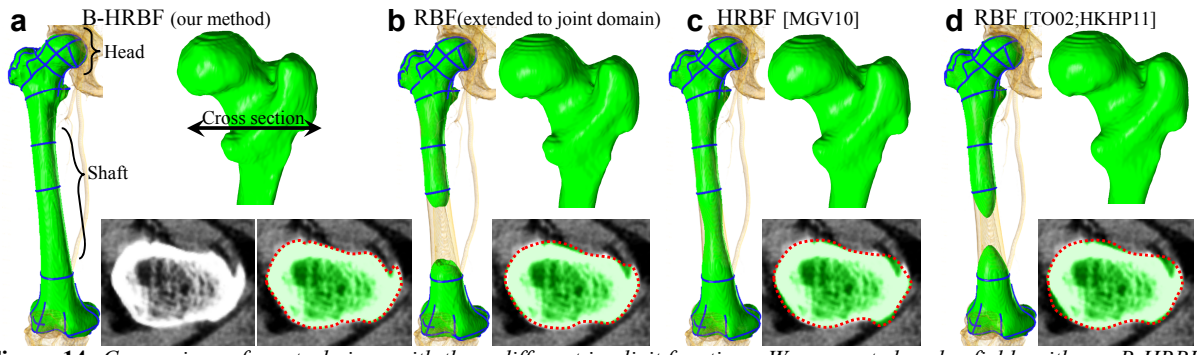

Figure 14: *Comparison of our technique with three different implicit functions. We computed scalar fields with our B-HRBF (a), an extended RBF (b), the HRBF [MGV10] (c), and the RBF [TO02; HKHP11] (d), and extracted their zero-level sets as boundaries. In (b), we simply extended the RBF [TO02; HKHP11] to the joint domain by using Eqs. (1) and (2). The scalar fields in (a, b) are computed in the joint domain, and those in (c, d) in the spatial domain. For the RBF-based methods in (b, d), we generated two additional points for each data point, as in Figure 2b with ε=0.5v, where v is the voxel pitch.*

model	# of data points stored contour	# of data points activated contour	# of voxels on boundary	Solve Eq. (8) with our recycling LU (sec)	Solve Eq. (8) without our recycling LU (sec)	evaluate B-HRBF (sec)
Fig. 9a	99(3)	60	20501	0.22	0.26	0.73
Fig. 9b	304(10)	24	18834	0.48	1.89	0.76
Fig. 9c	365(12)	24	29501	0.46	3.27	1.02
Fig. 9d	555(19)	77	50958	2.75	12.92	1.51

Table 1: *Performance.The columns contain the number (#) of data points related to the stored contours (# of stored contours), # of data points related to the activated contour, # of foreground voxels existing on the bundary, time required to solve the linear system in Eq. (8) with our recycling LU-decomposition scheme, time requred to solve the linear system without the recycling scheme, and time required to evaluate the scalar field via the surface tracking marching cubes.*

slightly missed the correct boundary because they did not consider image features (see the cross-sections in Fig. 14c, d). It is usually labor intensive to correct such small errors. Around the shaft, where contours are sparsely specified, only our B-HRBF generated a correct boundary. The methods using the RBF resulted in separated surfaces (Fig. 14b, d).

Table 1 shows the performance of our technique. All timings were generated with an Intel Core i7 3.33-GHz computer. The computational time depended on the segmentation target and the number of contours. We measured the time required to complete the four examples shown in Figure 9a-d. As Table 1 indicates, our recycling LU decomposition scheme drastically accelerated the B-HRBF computation. Although the total computation took a few seconds when many contours were placed, it was fast enough not to disturb the user's interactions. In addition, segmentation was computed on a background thread so that the user could edit contours without waiting.

7. Conclusions

We have presented a new contour-based user interface, a novel B-HRBF formulation for volume segmentation, and an acceleration scheme (i.e., recycling LU decomposition) for computing the B-HRBF. By combining the B-HRBF and the acceleration scheme, we have achieved real-time and intuitive volume segmentations for real-world volume images. Because our technique allows a user to directly place contours on ambiguous image areas, it is especially useful for extracting ROIs whose boundaries are difficult to detect using automatic algorithms. In our experiments, the B-HRBF achieved better segmentation boundaries than did previous contour-based techniques.

One limitation of our technique is that it is difficult for novice users to place cross-sections to specify contours. Well-distributed cross-sections may enhance segmentation, whereas poorly distributed ones may cause unnecessary contour specification. In the future, we will attempt to automatically generate good arrangements of cross-sections and suggest them to users. In this work, we generated a scalar field in the joint spatial-range domain (Euclidean space). Generating a scalar field directly on the image manifold (Riemann space) would be an interesting future direction. Our future work will also include investigation of automatic parameter tuning and integration of smart 2D contouring tools (e.g. livewire, magnetic lasso, or intelligent scissors) with our contour-based interface. In addition, we would like to tackle the problem of integrating our contour-based interface with level-set-based image segmentation methods [GBFP07; XM11].

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments. We also thank Dr. Yutaka Ohtake for his comments based on his deep insights. This work was supported in part by the Grant-in-Aid for Scientific Research of Japan (24700182 and 20113007) and the National Cancer Center Research and Development Fund (23-A-26). The stuffed bear CT volume is courtesy of [VL13].

References

[AB94] ADAMS, R. BISCHOF, L.: Seeded region growing. *IEEE Trans*. *PAMI* 16, 6(1994), 641–647.

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

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new voronoi-based surface reconstruction algorithm. In *Proc. SIGGRAPH 1998*, 415–422.

[BDH96] BARBER C.B., DOBKIN D.P., HUHDANPAA H.T.: The Quickhull algorithm for convex hulls. *ACM Trans. on Math. Soft. 22*, 4(1996), 469–483.

[BMR*99] BERNARDINI F., MITTLEMAN J., RUSHMEIER H., SILVA C., TAUBIN G.: The Ball-Pivoting algorithm for surface reconstruction. *IEEE TVCG* 5, 4(1999), 349–359.

[BK05] BOTSCH M., KOBBELT L.: Real-time shape editing using radial basis functions. *CGF* 24, 3(2005), 611–621.

[BK03] BOYKOV, Y., KOLMOGOROV, V.: Computing geodesics and minimal surface via graph cuts. In *Proc. ICCV 2003*, I, 26–33.

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

[BMS*10] BRAZIL E. V., MACÊDO I., SOUSA M. C., DE FIGUEIREDO L. H., VELHO L.: Sketching variational hermite-RBF implicits. In *Proc. SBIM 2010*, 1–8.

[BDP*05] BRUIN P.W., DERCKSEN V.J., POST F.H., VOSSEPOEL A.M., STREEKSTRA G.J., VOS F.M.: Interactive 3D segmentation using connected orthogonal contours. *Comp. Biol. Med.* 35, 4(2005), 329–346.

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *Proc. SIGGRAPH* 2001, 67–76.

[Duc77] DUCHON J.: Splines minimizing rotation-invariant seminorms in Sobolev spaces, Constructive Theory of Functions of Several Variables. *Lect. Notes in Math.*, 571, Springer, Berlin, 1977, 85–100.

[EM94] EDELSBRUNNER H., MÜCKE E. P.: Three dimensional alpha shapes. *ACM TOG* 13, 1(1994), 43–72.

[GBFP07] Gelas A. Bernard O., Friboulet D., Prost R.: Compactly Supported Radial Basis Functions Based Collocation Method for Level-Set Evolution in Image Segmentation. *IEEE TIP* 16, 7(2007), 1873-1887.

[Gon92] Gondzio J.: Stable algorithm for updating dense LU factorization after row or column exchange and row and column addition or deletion. *Optimization: J. Math. Prog. Oper. Res. 23, 1(1992), 7-26.*

[Gra04] GRANT J.C.B.: *Grant's Atlas of Anatomy* (11th Edition). Williams & Wilkins, 2004.

[HKHP11] HECKEL F., KONRAD O., HAHN H. K., PEITGEN H. O.: Interactive 3D medical image segmentation with energy-minimizing implicit function. *VCBM* 35, 2(2011), 275–287.

[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. *IVC* 26, 2(2008), 141–163.

[IY10] IJIRI T., YOKOTA H.: Contour-based interface for refining volume segmentation. *CGF* 29, 7(2010), 2153–2160.

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

[KHR02] KARPENKO O., HUGHES J. F., RASKAR R.: Free-form sketching with variational implicit surfaces. *CGF*, 21, 3(2002), 585–594.

[LSTS04] LI Y., SUN J., TANG C. K., SHUM H. Y.: Lazy snapping. *ACM TOG* 23, 3(2004), 303–308.

[LBD*08] LIU L., BAJAJ C., DEASY J. O., LOW D. A., JU T.: Surface reconstruction from non-parallel curve networks. *CGF* 27, 2(2008), 155–163.

[MGV10] MACÊDO I., GOIS J. P., VELHO L.: Hermite radial basis functions implicits. *CGF* 30, 1(2010), 27–42.

[OBS04] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: 3D scattered data approximation with adaptive compactly supported radial basis functions, In *Proc. SMI 2004*, 31–39.

[PD06] Paris S., Durand F.: A Fast Approximation of the Bilateral Filter using a Signal Processing Approach. In *Proc. ECCV* 2006, 568-580.

[PXP98] PHAM D. L., XU C., PRINCE J. L.: A survey of current methods in medical image segmentation. In *Tech. Rep. JHU/ECE 99–01*, *Johns Hopkins Univ.*, 1998.

[PTVF07] PRESS W. H., TEUKOLSKY S. A., VETTERLING W.T., FLANNERY B.P.: *Numerical Recipes (3rd Edition): The Art of Scientific Computing*. Cambridge Univ. Press, 2007.

[SFYC96] Shekhar F., Fayyad E., Yagel R., Cornhill J.F.: Octree-Based Decimation of Marching Cubes Surfaces, In *Proc. of IEEE Vis.* (1996), 335–342.

[SKM98] SOCHEN N., KIMMEL R., MALLADI, R.: A general framework for low level vision. *IEEE TIP*. 7(1998), 310–318.

[TM98] TOMASI, C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Proc. ICCV*, 1998, 836–846.

[TO02] TURK G., O'BRIEN J. F.: Modelling with implicit surfaces that interpolate. *ACM TOG* 21, 4(2002), 855–873.

[VL13] The Volume Libraly, http://lgdv.cs.fau.de/External/vollib/

[WSC06] WALDER C., SCHÖLKOPF B., CHAPELLE O.: Implicit surface modelling with a globally regularised basis of compact support. *CGF* 25, 3 (2006), 635–644.

[Wen05] WENDLAND H.: *Scattered Data Approximation*. Cambridge Univ. Press, Cambridge, 2005.

[Wir07] WIRJADI O.: Survey of 3D image segmentation methods. *ITWM Rep.*, 123, 2007.

[XM11] Xie X., Mirmehdi M.: Radial basis function based level set interpolation and evolution for deformable modeling. *IVC* 29, 2-3, 2011, 167-177.