3D Segmentation of Soft Organs by Flipping-Free Mesh Deformation *

Feng Ding
Dept. of Computer Science
National University of Singapore
dingfeng@comp.nus.edu.sg

Wenxian Yang
School of Computer Engineering
Nanyang Technological University
wxyang@ntu.edu.sg

Wee Kheng Leow
Dept. of Computer Science
National University of Singapore
leowwk@comp.nus.edu.sg

Sudhakar K Venkatesh
YLL School of Medicine
National University of Singapore
dnrskv@nus.edu.sg

Abstract

Segmentation of 3D soft organs from complex volume images is a very important and challenging task. The objects of interest may have inhomogeneous voxel intensities and some object boundaries may be indistinct. Existing algorithms are either sensitive to noise or computationally expensive. This paper presents a novel algorithm that overcomes these shortcomings. The algorithm adopts a novel flipping-free mesh deformation and registration method that can easily incorporate geometric constraints to reduce sensitivity to noise. It efficiently deforms the 3D model in large displacements reducing total computational costs. These properties are confirmed by comprehensive test results.

1. Introduction

Segmentation of 3D soft organs from CT and MR is a very important and challenging task for medical image analysis. The objects may have inhomogeneous voxel intensities and some object boundaries may be indistinct. Segmentation methods such as thresholding, region growing, watershed and classification work well on simple images with homogeneous regions. Unfortunately, they are sensitive to noise and produce severe over-segmentation when applied to medical images with inhomogeneous regions.

Interactive segmentation algorithms such as GrabCut [18] and random walks [7] achieve fairly good results in 2D color images. However, they are computationally expensive in both time and space especially for 3D medical data, which often contains more than $512 \times 512 \times 200$ voxels. The memory usage of graph cut, for instance, may exceed 4GB for such data set, which is beyond the limit of 32-bit computers. Moreover, these algorithms may produce foreground regions with undesirable topology. A recent implementation [4] reduces the memory requirement slightly.

Deformable models have been successfully applied to medical image segmentation. They can be represented either implicitly or explicitly. Segmentation methods using implicit models such as the level set method [20] and the fast marching methods [21] represent a 3D surface as an implicit function discretized into voxels, resulting in computationally expensive algorithms. The level set method can change the evolving surface's topology to match highly complex object surface. However, it often leaks out of the object boundaries producing undesired segmentation.

In contrast, segmentation methods using explicit models [2, 16] represent a 3D surface as a mesh, which significantly reduce the space complexity of the algorithms. Deformation is accomplished by displacing the mesh vertices. The problem of mesh-based methods is that the displacements of vertices may cause self-intersections of the mesh, which can be categorized as *flipping* or *non-flipping*. Flipping self-intersection occurs locally if the displacement vectors of neighboring mesh vertices cross in space. As a result, the directions of some surface normals flip after deformation. This problem cannot be solved by simply reducing the deformation step size. As shown in Fig. 3, surface flippings occur during mesh deformation towards a binary volume data even with a very small deformation step size. Non-flipping self-intersection occurs globally without flipping the surface normals but causes penetration of different parts of the mesh.

In summary, existing 3D deformable model-based algorithms exhibit various weaknesses that need to be overcome. To address these problems, we propose an algorithm for segmenting 3D soft organs such as liver and spleen based

*This research is supported by SBIC RP C-008/2006 and A*STAR SERC 052 101 0103 (NUS R-252-000-319-305).

on mesh deformation. The algorithm is efficient, noise resilient, topology preserving and free from flipping self-intersection. Geometric constraints are easily incorporated into the model to improve noise resilience and reduce the likelihood of non-flipping self-intersection (Section 3).

2. Related Work

Segmentation methods based on mesh deformation can be categorized according to their strategies for handling self-intersections.

2.1. No Explicit Handling of Self-intersections

Deformable models such as active contour [12], mass spring model [5] and Laplacian deformation [23] do not explicitly handle self-intersections. They impose smoothness constraints on the mesh, which can alleviate self-intersections to some extent. Nevertheless, self-intersection can still happen [10]. Active shape model [2] used by many segmentation algorithm [6, 22, 24] generates new shapes without self-intersection by varying the shape coefficients derived from PCA. However, a large number of training samples are required to derive a complex model. The method in [17] uses restricted Delaunay triangulation to re-triangulate the mesh in each iteration based on extracted image features. It is thus computationally expensive and sensitive to noise.

2.2. Detection of Self-intersections

The T-snake model [15] discretizes the space underlying the mesh into grids and detects self-intersections after deformation by inspecting the local neighborhoods of the grid points. It resolves self-intersections by falling back to the non-intersecting state. The method in [14] imposes proximity conditions between mesh vertices. By displacing mesh with a very small step size, violation of proximity conditions is detected, and the model is remeshed to remove self-intersections. The methods in [3, 11, 26] detect self-intersections based on collision detection and resolve them by remeshing. In general, collision detection and remeshing are computationally expensive, and they contribute to most of the computational costs of the algorithms.

2.3. Avoidance of Self-intersections

Under Free-Form Deformation (FFD) [19], self-intersections can be avoided by imposing injectivity condition [1, 8] on the deformation function. The injectivity condition confines the displacements of FFD control points within regions that do not incur self-intersections. For segmentation, directly displacing mesh vertices, as in Directly Manipulated FFD [9], is preferred so that the mesh surfaces can be accurately aligned to the target boundaries. Unfortunately, it is nontrivial to derive the injectivity condition of mesh vertices from that of the control points. Moreover, the injectivity condition limits the displacements of control points to short ranges, resulting in very slow convergence. Another approach is to compute a diffeomorphic deformation function [13, 27]. As a diffeomorphic function and its inverse are one-to-one and smooth, self-intersections are avoided. However, computation of the diffeomorphic function is expensive, especially when it is applied to the segmentation of complex and noisy 3D volume images.

3. Segmentation by Deformable Registration

Our algorithm, as summarized in Algorithm 1, segments a 3D object from an input volume by iteratively deforming a 3D mesh model to register it to extracted image features. In each iteration, it searches for possible correspondence between mesh vertices and image features over long distances. The detected correspondence is refined before deformation to avoid flippings. To clarify the algorithm, we first introduce the 3D quadrilateral mesh model.

3.1. 3D Quadrilateral Mesh

We define a 3D quadrilateral mesh M on a cube whose sides are aligned with x-, y- and z-axis (Fig. 1(a)). The mesh is defined by 3 groups G_{xy}, G_{yz}, and G_{zx} of closed contours that are parallel to the xy-, yz- or zx-plane respectively. The contours in a group are *orthogonal* to those in the other groups. Each vertex $\mathbf{u}_i$ in M is an intersection of two contours, each from a different group. Thus, it has exactly 4 connected neighboring vertices. The cubical quadrilateral mesh can be mapped to a spherical quadrilateral mesh (Fig. 1(b)) by projecting its mesh vertices onto a concentric spherical surface along the radius directions.

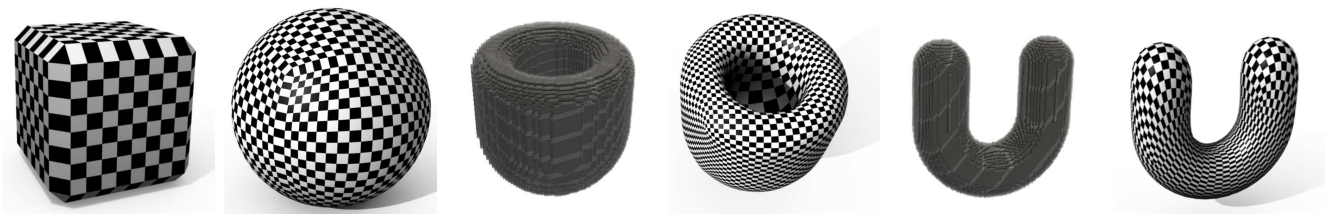
The proposed mesh has the advantage that a linear ordering of the mesh vertices can be defined along any closed contour. As will be discussed in Section 3.5, the linear ordering simplifies the avoidance of possible flippings. In comparison, avoidance of flippings in a triangular mesh is much more difficult to achieve because it is difficult to define linear ordering of all the vertices.

3.2. Image Feature Extraction

Our segmentation algorithm can work with any feature including, but not restricted to, intensity, gradient, edge,

Algorithm 1 Segmentation by deformable registration.

Extract image features from target volume (Section 3.2).
Repeat until convergence:
  Search for correspondence (Section 3.3).
  For each contour in each group
    Detect possible flippings (Section 3.4).
    Avoid possible flippings (Section 3.5).
  Perform mesh deformation (Section 3.6).



(a) (b) (c) (d) (e) (f)

Figure 1. The cubical quadrilateral mesh (a) is projected onto a spherical surface to generate (b) a spherical quadrilateral mesh. Registration of (b) to concave target volumes (c, e) produces (d, f). Checker board patterns are added for visual clarity.

texture, etc. We proposed to use voxel intensity distribution because the objects have inhomogeneous voxel intensities and some object boundaries are indistinct. Intensities of the foreground object are modeled as a mixture of Gaussians:

$$g(x) = \sum_i a_i f_i(x), \tag{1}$$

where x is the voxel intensity, a_i are coefficients, such that $\sum_i a_i = 1$, and $f_i(x)$ are Gaussian distributions with parameters (μ_i, σ_i). The number of Gaussians is determined by the input images and target organs. Parameters a_i, μ_i and σ_i can be estimated by Expectation Maximization (EM). To smooth out noise, anisotropic filtering is applied to the input image as a pre-processing step.

3.3. Correspondence Search

This stage searches for correspondence between the model M and the target T. For each vertex $\mathbf{u}_i$ on M, the algorithm searches along the projection line $P(\mathbf{u}_i)$, which can be defined as the surface normal at $\mathbf{u}_i$, for a possible corresponding point $\mathbf{v}_i$ on the surface of T. The point $\mathbf{v}_i$ is the intersection of $P(\mathbf{u}_i)$ and the face of a feature voxel on the surface of T. Each $\mathbf{v}_i$ serves as a target location for $\mathbf{u}_i$. In general, $P(\mathbf{u}_i)$ may be defined along other appropriate directions. $\mathbf{u}_i$ is labeled as a *solitary* vertex if its corresponding point cannot be found.

The problem of finding correspondence $\mathbf{u}_j$ for $\mathbf{u}_i$ along $P(\mathbf{u}_i)$ is formulated as finding the minimum j such that

$$\sum_{i=j}^{j+N} h(\mathbf{u}_i) = 0, \quad h(\mathbf{u}_i) = \begin{cases} 0 & g(x_i) < \Gamma \\ 1 & \text{otherwise} \end{cases}, \tag{2}$$

where x_i is the intensity of $\mathbf{u}_i$ and Γ is a pre-defined threshold. $\mathbf{u}_j$ that satisfies Eq. (2) is likely on the boundary of the target object, since all $(N+1)$ consecutive voxels along $P(\mathbf{u}_i)$ starting form $\mathbf{u}_j$ have low probabilities of belonging to the foreground. In our current implementation, $N = 3$.

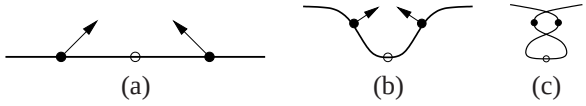
3.4. Flip Detection

The flipping of a mesh cell after mesh deformation is characterized by the flipping of at least one of its edges. Therefore, surface flipping can be identified by detecting edge flipping. Let $\mathbf{u}_i$ and $\mathbf{u}_j$ denote two non-solitary neighbors on a closed contour, and $\mathbf{v}_i$ and $\mathbf{v}_j$ denote their respective corresponding points on the target. Then, edge flipping occurs when the orientations of the edges $\mathbf{u}_i - \mathbf{u}_j$ and $\mathbf{v}_i - \mathbf{v}_j$ differ significantly:

$$\frac{\mathbf{u}_i - \mathbf{u}_j}{\|\mathbf{u}_i - \mathbf{u}_j\|} \cdot \frac{\mathbf{v}_i - \mathbf{v}_j}{\|\mathbf{v}_i - \mathbf{v}_j\|} \leq \tau, \tag{3}$$

where $\tau \in [0, 1)$ is a predefined threshold. The vertices $\mathbf{u}_i$ and $\mathbf{u}_j$ that form a flipping edge are labeled as *flipping* vertices; otherwise, *non-flipping* vertices.

As each vertex $\mathbf{u}_i$ is an intersection of two orthogonal closed contours on M, flipping may occur along both contours Therefore, each $\mathbf{u}_i$ will undergo flip detection along both contours when the algorithm iterates.



(a) (b) (c)

Figure 2. Folding problem. (a) Displacing non-flipping vertices (dots) around solitary vertices (circle) may cause (b) folding of the mesh, and in the extreme case, (c) non-flipping self-intersection.

3.5. Flip Avoidance

Our method avoids flippings by discarding the point correspondences of flipping vertices. Let $\mathbf{u}_i, \mathbf{u}_{i+1}, \ldots, \mathbf{u}_n$ denote a consecutive sequence of flipping vertices on a closed contour, excluding solitary vertices, such that $\mathbf{u}_{i-1}$ and $\mathbf{u}_{n+1}$ are non-flipping. The method identifies the middle flipping vertex $\mathbf{u}_m$ of the sequence, labels it as non-flipping, and labels the other flipping vertices as solitary, i.e., discarding their correspondences. After repeating this process for every closed contour, only non-flipping vertices have point correspondences. Thereafter, deforming the mesh according to these correspondences does not result in flipping.

3.6. Deformation

During mesh deformation, if non-flipping vertices are displaced to their target locations while solitary vertices remain undisplaced (Fig. 2(a)), the mesh may fold around

solitary vertices (Fig. 2(b)), and in an extreme case, results in non-flipping self-intersections (Fig. 2(c)). To tackle this problem, the displacement vectors of non-flipping vertices are propagated to neighboring solitary vertices, turning them into non-flipping vertices, by iterative local averaging of displacement vectors. This process is analogous to the diffusion of gradient vectors in [25]. It also smoothens the variation of displacement vectors among neighboring non-flipping vertices, thus improving noise resilience.

The Laplacian method [23] is adopted for mesh deformation because it is very efficient, easy to use, and easy to incorporate geometric constraints. During deformation, non-flipping vertices are displaced towards their target locations, which are regarded as *positional constraints*. The other mesh vertices are displaced according to geometric constraints including the preservation of Laplacians (i.e., curvature normals) and *uniform vertex distribution*. The deformation problem is formulated as minimizing the energy

$$E = E_L + \lambda_p E_p + \lambda_u E_u \tag{4}$$

where E_L, E_p and E_u are energies for Laplacian preservation, positional and uniform vertex distribution, and λ_p and λ_u are weighting parameters. E_L is defined as:

$$E_L = \sum_i \|L(\mathbf{u}_i) - L(\mathbf{u}'_i)\|_2^2, \tag{5}$$

where $\mathbf{u}'_i$ denotes the position of vertex $\mathbf{u}_i$ after deformation. L denotes the Laplacian operator,

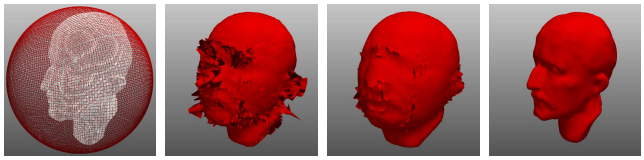
$$L(\mathbf{u}_i) = \mathbf{u}_i - \frac{1}{|\mathcal{N}(i)|} \sum_{\mathbf{u}_j \in \mathcal{N}(i)} \mathbf{u}_j, \tag{6}$$

where $\mathcal{N}(i)$ denotes neighboring vertices of $\mathbf{u}_i$ and $|\mathcal{N}(i)|$ denotes the number of the neighboring vertices. Minimizing E_L results in local shape preservation and mesh smoothness. The positional energy E_p imposes the positional constraint, and is defined as:

$$E_p = \sum_i \frac{d_i}{\bar{d}} \|\mathbf{u}_i - \mathbf{p}_i\|_2^2, \tag{7}$$

where $\mathbf{p}_i$ denotes the target position (positional constraint) of $\mathbf{u}_i$, d_i is the distance from $\mathbf{u}_i$ to $\mathbf{p}_i$, $\bar{d}$ is the average of d_i for $\mathbf{u}_i$ of M. Minimizing E_p displaces mesh vertices towards their designated locations. In addition, positional constraints are weighted according to the distance between $\mathbf{u}_i$ and $\mathbf{p}_i$, i.e., $d_i/\bar{d}$. The further the corresponding point, the larger is the weight. This weighting scheme ensures that the mesh vertices are displaced mainly along the surface normals so that the model can deform towards the target surface quickly. E_u is defined as:

$$E_u = \sum_C \sum_{\mathbf{u}_i, \mathbf{u}_j \in C}^{\mathbf{u}_j \in \mathcal{N}(i)} \|(\mathbf{u}_i - \mathbf{u}_j) - \frac{\bar{l}}{l_{ij}}(\mathbf{u}'_i - \mathbf{u}'_j)\|_2^2, \tag{8}$$



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

Figure 3. Comparison of naive and flipping-free registration. (a) Initialization for a binary volume image. (b) Naive method with $\lambda = 0.3$ and 4 iterations. (c) Naive method with $\lambda = 0.01$ and 100 iterations. In (b, c), surface flippings cause discontinuities. (d) Proposed algorithm with $\lambda = 0.3$ and 20 iterations.

where $\bar{l}$ is the estimated average distance between $\mathbf{u}'_i$ and $\mathbf{u}'_j$, and l_{ij} is the current edge length between $\mathbf{u}_i$ and $\mathbf{u}_j$ on a contour C. Minimizing E_u enables the vertices to distribute more evenly on the mesh surface. When the mesh vertices are close to the target surface, the weights of positional constraints become small. At the same time, the constraint of uniform vertex distribution becomes significant, and it minimizes the difference between the length l_{ij} of a mesh edge and the average length $\bar{l}$, which is estimated from the length of the closed contour that includes the edge. The energy E_u reaches the minimum value 0 when all l_{ij} are equal to $\bar{l}$, i.e., all mesh edges are of the same length. Thus, the mesh vertices can displace along directions tangential to the mesh surface and distribute more evenly.

The non-linear total energy E is minimized using Gauss-Newton iteration with the efficient Taucs solver.

In summary, the proposed algorithm can efficiently detect and avoid possible flippings before each deformation iteration. Segmentation is achieved by efficient Laplacian deformation with various constraints. The strength of our algorithm will be discussed in the next section.

4. Experiments and Discussions

Comprehensive tests were conducted to verify the strength of our algorithm in terms of flip detection and avoidance (Section 4.1), convergence and noise resilience (Section 4.2), accuracy and efficiency (Section 4.3). The experiments were mainly carried out on globular objects such as liver and spleen. Applicability of the proposed algorithm to tubular objects is also discussed in Section 4.4. All experiments were performed on an Intel Core 2 Duo 2.33 GHz computer with 4G memory.

4.1. Flip Avoidance

To understand the importance of flip detection and avoidance in mesh deformation-based segmentation, mesh registration to a binary volume image without flip detection and avoidance was performed. As shown in Fig. 3(b, c), surface discontinuities caused by flippings occur even with a very small deformation step size. In comparison, registration using our proposed algorithm is flipping-free (Fig. 3(d)).

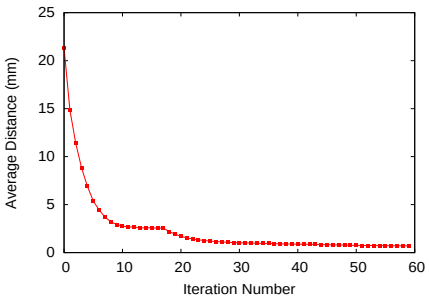


Figure 4. Convergence curve. The average estimated distance from model vertices to target points decreases as the algorithm iterates.

4.2. Convergence and Noise Resilience

This experiment shows that our algorithm can converge in registering the model to concave objects. The initial spherical model (Fig. 1(b)) was manually placed outside the volume images of a cup object (Fig. 1(c)). The model converged to the target surface (Fig. 1(d)) in 150 iterations.

To demonstrate convergence and noise resilience properties of the proposed algorithm in real medical volume images, segmentation of soft organs from abdominal CT scans with slice thickness of 1mm to 3mm was carried out. The level set algorithm and the proposed algorithm were both performed on 8 CT scans. Graph cut was performed only on a single slice from one scan because it is an interactive algorithm and can achieve arbitrarily high accuracy given substantial amount of user interaction. In our experiment, result from graph cut was obtained by moderate user input (Fig. 5(i)). The target organs were the livers and the spleens. The model was initialized as a spherical mesh (Fig. 5(a)) since the targets were globular. Voxels inside the sphere were used to build a Gaussian mixture model (GMM) of the intensity probability distribution of the target organs (foreground). Voxels with low probability were regarded as background feature voxels. Transitions from consecutive foreground voxels to consecutive background voxels along searching directions suggest the presence of boundary points of the target organ. Figure 5(f, g) show two views of the segmented liver using our algorithm, which has a complex shape.

To demonstrate the convergence property of our segmentation algorithm, the average distance from the model vertices to the target points over each iteration for one test data is plotted (Fig. 4). The distance decreases rapidly as the algorithm iterates. Convergence of our algorithm was also confirmed by other test data with similar descending curves.

The proposed algorithm is noise resilient because it looks for correspondence over a long range, and is thus less affected by local noise. It also incorporates geometric constraints that increase its resilience to local noise. In comparison, segmentation using the snake algorithm was trapped by local noise, as shown in Fig. 5(e).

4.3. Accuracy and Efficiency

Our algorithm was compared with GVF snake [25], level set method as implemented in ITK-SNAP and graph cut [18] in terms of noise resilience, accuracy and efficiency.

Our algorithm and level set were applied to the whole CT volume and initialized with spheres of the same size at the same location (Fig. 5(a)). GVF snake was applied on a single CT slice because of its 2D nature. It was initialized with a 2D cross-section of the sphere in the slice. Graph cut was applied to a single CT slice because its 3D implementation ran out of memory given the input volume images. It was initialized by manual markups representing foreground and background pixels (Fig. 5(i)). The level set algorithm was stopped immediately by the user when the liver regions were fully segmented.

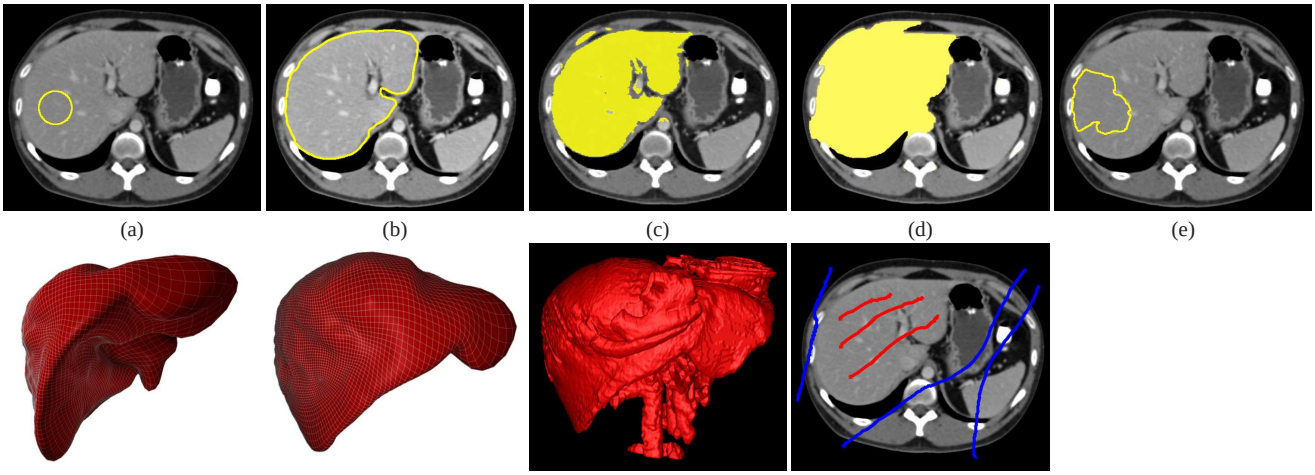
As shown in Fig. 5(h) and (d), indistinct boundaries caused severe leakage problem for both level set and graph cut. The noise problem prevented the GVF snake from converging to the target boundary (Fig. 5(e)). In comparison, our algorithm has less leakage thanks to the geometric constraints. Segmentation accuracy in terms of average symmetric distance and volume overlap was computed for level set and our algorithm (Table 1).Our algorithm achieved better accuracy with shorter average symmetric distance and larger volume overlap. The much lower variance achieved by our algorithm also indicates it is more stable.

With regard to efficiency, the level set algorithm took 1051 iterations in 476.29 seconds on average to segment the whole liver. In contrast, our algorithm took only 43 iterations in 54.47 seconds on average to segment the liver. Compared to graph cut, our algorithm also took much less time to segment a slice. Note that the level set algorithm implemented in ITK-SNAP automatically used two threads for computation in our PC, whereas graph cut and our algorithm used one thread only.

Segmentation was also performed on another abdominal organ, i.e., spleen. As shown in Table 1, the results produced by level set (Fig. 6(d) top) is less smooth than those produced by our algorithm (Fig. 6(c) top) due to voxelization, and they have leakage artifacts. Our algorithm performed more accurately than level set in segmenting the spleen. Execution time for a slice was also computed across different algorithms. The results in Table 1 show that the proposed algorithm is much faster than the level set algorithm and graph cut.

4.4. Segmentation of Tubular Organ

To further test the capability of the proposed algorithm to segment target organs with various shapes, registration of the model to tubular target, e.g., a U-shaped volume (Fig. 1(e)) is performed. The initial spherical mesh was outside of the target, and it successfully converged to the target



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

(f) (g) (h) (i)

Figure 5. Comparison of segmentation algorithms. (a) Initialization for proposed algorithm, level set and GVF snake. Segmentation results of (b, f, g) proposed algorithm, (c, h) level set, (d) graph cut, (e) GVF snake. (i) Initialization for graph cut: (red) foreground and (blue) background markups. Best viewed in color.

Table 1. Comparison of level set algorithm (LS), graph cut (GC) and the proposed algorithm. V: ground truth volume. D: average symmetric distance. VO: volume (area for graph cut) overlap. K: number of iterations. T: execution time. T′: execution time per slice.

	V (mm^3)	Algorithm	D (mm)	VO	K	T (sec)	T′ (sec)
liver	1754.37±387.57	LS	10.28±3.25	72.98±8.12%	1051±138	476.29±116.27	3.58±2.46
		GC	10.49	81.50%	23	17.30	17.30
		proposed	1.69±0.40	89.97±1.52%	43±6	54.47±7.38	0.42±0.26
spleen	367.89±196.43	LS	4.43±3.59	76.10±15.06%	519±75	110.38±43.60	0.71±0.27
		GC	0.82	96.00%	13	11.56	11.56
		proposed	1.00±0.27	88.87±2.98%	42±11	17.84±4.50	0.14±0.11
left brachiocephalic vein	6.05	LS	0.38	81.20%	387	53.04	0.53
		GC	1.36	81.20%	14	10.99	10.99
		proposed	0.35	83.00%	72	26.01	0.26

surface in 150 iterations (Fig. 1(f)).

Segmentation of left brachiocephalic vein which has a tubular shape from real medical images was performed. The initialization was inside the blood vessel (Fig. 6(a) bottom). The left brachiocephalic vein was successfully segmented (Fig. 6(b, c) bottom) thanks to the uniform vertex distribution constraint, which facilitated the large shape change. Results were compared with those obtained by the level set methods and graph cut (Table 1). Segmentation accuracy using our algorithm is only slightly better than that of level set and graph cut because the object has a very small volume ($6mm^3$) and a segmentation error of a single voxel will result in large error in volume overlap. Again, our algorithm is more efficient.

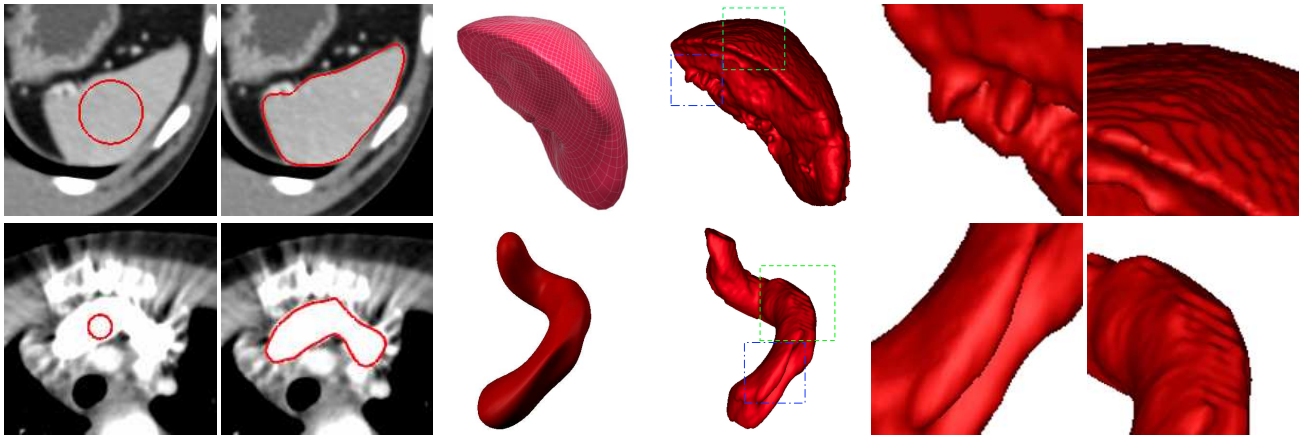
5. Conclusions

This paper presented a 3D volume image segmentation algorithm based on a novel flipping-free mesh deformation method. The proposed algorithm is free from flipping self-intersection by detecting and avoiding possible flippings efficiently before each deformation iteration. It also alleviates folding and non-flipping self-intersection problems by propagating displacement vectors. Our algorithm is noise resilient because (1) it looks for possible feature points over a long range instead of within a small local neighborhood, and (2) it incorporates geometric constraints. It is very efficient thanks to explicit mesh representation and the Laplacian deformation algorithm. Comprehensive tests show that it is more noise resilient, accurate and efficient than snake, level set and graph cut. Future work may include handling of global self-intersection in some extreme cases and application to vessel tree segmentation.

References

[1] Y. Choi and S. Lee. Injectivity conditions of 2D and 3D uniform cubic b-spline functions. *Graphical Models*, 62(6):411–427, 2000.

[2] T. F. Cootes, A. Hill, C. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12(6):355–366, 1994.



(a) (b) (c) (d) (e) (f)

Figure 6. Segmentation of (top) spleen and (bottom) left brachiocephalic vein. (a) Initialization. (b, c) Segmentation results of the proposed algorithm. (d) Segmentation results of level set. (e, f) Zoom-in views of leakages and artifacts in (d).

[3] H. Delingette and J. Montagnat. Shape and topology constraints on parametric active contours. *Computer Vision and Image Understanding*, 83(2):140–171, 2001.

[4] A. Delong and Y. Boykov. A scalable graph-cut algorithm for N-D grids. In *CVPR*, pages 1–8, 2008.

[5] L. Domheim, K. D. Tonnies, and J. Dornheim. Stable dynamic 3D shape models. In *ICIP*, volume 3, pages 1276–1279, 2005.

[6] J. Fripp, S. Crozier, S. Warfield, and S. Ourselin. Automatic segmentation of articular cartilage in magnetic resonance images of the knee. In *MICCAI*, pages 186–194, 2007.

[7] L. Grady. Random walks for image segmentation. *PAMI*, 28(11):1768–1783, 2006.

[8] M. Hagenlocker and K. Fujimura. CFFD: a tool for designing flexible shapes. *The Visual Computer*, 14:271–287, 1998.

[9] W. M. Hsu, J. F. Hughes, and H. Kaufman. Direct manipulation of free-form deformations. In *ACM SIGGRAPH*, pages 177–184, 1992.

[10] L. Ji and H. Yan. Robust topology-adaptive snakes for image segmentation. *Image and Vision Computing*, 20:147–164, 2002.

[11] W. Jung, H. Shin, and B. K. Choi. Self-intersection removal in triangular mesh offsetting. In *Computer-Aided Design and Applications*, pages 477–484, 2004.

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4):321–331, 1988.

[13] A. Khan, E. Aylward, P. Barta, M. Miller, and M. F. Beg. Semi-automated basal ganglia segmentation using large deformation diffeomorphic metric mapping. In *MICCAI*, pages 238–245, 2005.

[14] J.-O. Lachaud and A. Montanvert. Deformable meshes with automated topology changes for coarse-to-fine three-dimensional surface extraction. *Medical Image Analysis*, 3(2):187–207, 1998.

[15] T. McInerney and D. Terzopoulos. Topology adaptive deformable surfaces for medical image volume segmentation. *TMI*, 18(10):840–850, 1999.

[16] T. McInerney and D. Terzopoulos. T-snakes: Topology adaptive snakes. *Medical Image Analysis*, 4:73–91, 2000.

[17] J.-P. Pons and J.-D. Boissonnat. Delaunay deformable models: Topology-adaptive meshes based on the restricted delaunay triangulartion. In *CVPR*, pages 1–8, 2007.

[18] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH*, pages 309–314, 2004.

[19] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *ACM SIGGRAPH*, pages 151–159, 1986.

[20] J. A. Sethian. *Level Set Methods*. Cambridge University Press, 1996.

[21] J. A. Sethian. Fast marching methods. *SIAM review*, 41(2):199–235, 1999.

[22] Y. Shi and D. Shen. Hierarchical shape statistical model for segmentation of lung fields in chest radiographs. In *MICCAI*, pages 417–424, 2008.

[23] O. Sorkine, Y. Lipman, D. Cohen-Or, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Eurographics*, pages 179–188. ACM Press, 2004.

[24] S. Viswanath, B. Bloch, E. Genega, N. Rofsky, R. Lenkinski, J. Chappelow, R. Toth, and A. Madabhushi. A comprehensive segmentation, registration, and cancer detection scheme on 3 tesla in vivo prostate DCE-MRI. In *MICCAI*, pages 662–669, 2008.

[25] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *Image Processing*, 7(3):359–369, 1998.

[26] A. Zaharescu, E. Boyer, and R. Horaud. Transformesh : A topology-adaptive mesh-based approach to surface evolution. In *ACCV*, pages 166–175, 2007.

[27] X. Zhuang, K. Rhode, S. Arridge, R. Razavi, D. Hill, D. Hawkes, and S. Ourselin. An atlas-based segmentation propagation framework using locally affine registration application to automatic whole heart segmentation. In *MICCAI*, pages 425–433, 2008.