

A Topology Preserving Deformable Model Using Level Sets

Xiao Han*, Chenyang Xu[†], and Jerry L. Prince*

*Center for Imaging Science, Electrical and Computer Engineering,
Johns Hopkins University, Baltimore MD 21218

[†] Siemens Corporate Research, Princeton NJ 08540
{xhan, prince}@iron.ece.jhu.edu; Chenyang.Xu@scr.siemens.com

Abstract

Active contour and surface models, also known as deformable models, constitute a class of powerful segmentation techniques. Geometric deformable models implemented via level-set methods have advantages over parametric ones due to their intrinsic behavior, parameterization independence, and ease of implementation. However, a long claimed advantage of geometric deformable models — the ability to automatically handle topology changes — turns out to be a liability in applications where the objects to be segmented have a known topology that must be preserved. In this paper, we present a geometric deformable model that preserves topology using the simple point concept from digital topology. This algorithm maintains the other advantages of standard geometric deformable models including sub-pixel accuracy and production of non-intersecting curves (or surfaces). Several experiments on simulated and real data are provided to demonstrate the performance of the proposed algorithm.

1. Introduction

Active contour and surface models, also called deformable models, are curves or surfaces that deform within two-dimensional (2D) or three-dimensional (3D) digital images under the influence of both internal and external forces and user defined constraints. Ever since their introduction by Kass et al. [1], these algorithms have been at the heart of one of the most active and successful research areas in edge detection, image segmentation, shape modeling, and visual tracking. Deformable models are classified as either parametric active contours (cf., [1–3]) or geometric active contours (cf., [4–8]) according to their representation and implementation. In particular, parametric active contours are represented explicitly as parameterized curves or surfaces in a Lagrangian formulation. Geometric active contours, on the other hand, are represented implicitly as level sets of higher-dimensional level set functions which evolve

according to an Eulerian formulation [9].

Parametric deformable models (PDMs) are the older of two formulations and have been used extensively in many applications (see [10], for example). Geometric deformable models (GDMs) were introduced more recently by Caselles et al. [4] and by Malladi et al. [5], and have several advantages. First, they are completely intrinsic and independent of the parameterization of the evolving contour. Thus, there is no need to add or remove nodes from the initial parameterization or adjust the spacing of the nodes. Second, the intrinsic geometric properties of the contour (e.g., curvature) can be easily determined from the level set function. Third, the propagating contour can automatically change topology (e.g., merge or split) without requiring an elaborate mechanism to handle such changes. Finally, the resulting curves or surfaces do not contain self-intersections, which are costly to prevent in parametric deformable models.

There are some applications in which the topology of the sought object is known, and the resulting deformable model should conform to this topology. For example, in the analysis of 3D brain images it is desirable that a reconstruction of the cortical surface have a topology that is consistent with brain anatomy [11, 12]. Recently, in fact, there have been several methods reported to correct the topology of a cortical segmentation after the initial (topologically wrong) segmentation [13, 14]. To enforce a given topology during evolution of the deformable model, parametric deformable models have always been used because their topology is explicitly maintained in the Lagrangian formulation. Self-intersections can become a problem, however, as simple curves and surfaces are generally required, and the computational demands related to self-intersection detection are very high [12]. Geometric deformable models prevent self-intersection, but there has been no mechanism prior to this paper to prevent topological changes during geometric curve/surface evolution.

Clearly, there is a need for geometric deformable models that enforce a topological constraint. In this paper, we develop such a topology preserving mechanism for geometric deformable models that guarantees that the final curve or

surface has exactly the same topology as the initial one. The topology preservation is achieved by maintaining the topology of the digital object enclosed by the implicit contour, for which we make use of the *simple point* criterion from digital topology [15, 16]. We note that our approach maintains the sub-pixel interpolation property of geometric deformable models, which sharply contrasts our method with the topology preserving region growing method of Mangin et al. [17]. The topology preserving mechanism we describe can be used with any existing 2D or 3D geometric deformable model, regardless of the internal or external force definition, yielding a large new class of topology preserving geometric deformable models.

2. Background

In this section, we first present a brief introduction to geometric deformable models and the evolution of level set functions. We then introduce some basic notions about digital topology and simple points, concepts that will be used to yield a topology preserving mechanism. We conclude with a description of isosurface and isocontour algorithms, which must be implemented correctly in order to yield an accurate and topologically correct representation of the implicit surface or contour extracted after topologically consistent evolution of its level set function.

2.1. Geometric deformable models Geometric deformable models are based on the theory of curve evolution and are implemented using the level set numerical method [9]. Let $I : U \rightarrow R^+$ be a given image, where $U \subset R^2$ in 2D and $U \subset R^3$ in 3D. In geometric deformable models, the evolving 2D curves or 3D surfaces are embedded as the zero level set of a higher dimensional level set function $\Phi(x, t) : U \times R^+ \rightarrow R$, and propagate implicitly through the temporal evolution of Φ . By convention, Φ is a signed distance function to the curves or surfaces with negative value inside the contour and positive outside. It can be computed efficiently by the fast marching method (cf., [9]).

The evolution of the level set function Φ is usually prescribed by a PDE of the following form (cf., [9]):

$$\Phi_t = F_{\text{prop}}|\nabla\Phi| + F_{\text{curv}}|\nabla\Phi| + F_{\text{adv}} \cdot \nabla\Phi, \quad (1)$$

where F_{prop} , F_{curv} and F_{adv} are speed or force terms that can be spatially varying. F_{prop} is an expansion or contraction speed. F_{curv} is a part of the speed that depends on the intrinsic geometry, especially the curvature κ of the contour and/or its derivatives. F_{adv} is an underlying velocity field that passively transports the contour. As an example, we can choose F_{prop} to be a region force (also known as a signed pressure force) $R(x)$, and F_{adv} to be a gradient vector flow force $\bar{v}(x)$ [3], and the evolution equation becomes:

$$\Phi_t(x) = \omega_R R(x)|\nabla\Phi| + \omega_\kappa \kappa(x)|\nabla\Phi| + \omega_{\bar{v}} \bar{v}(x) \cdot \nabla\Phi, \quad (2)$$

where ω_R , ω_κ , and $\omega_{\bar{v}}$ are the weights for the respective forces. For a 0-1 valued binary image I , $R(x)$ can be defined to be $R(x) = 2I(x) - 1$, so that $R(x)$ is an expansion force inside the object and a contraction force outside.

The numerical solution of Eq. (1) can be obtained by approximating the time derivative by a forward difference and the spatial derivatives on the right-hand side by upwind numerical schemes (for details see [9]), which gives:

$$\Phi^{m+1}(x) = \Phi^m(x) + \Delta t \Delta \Phi^m(x), \quad (3)$$

where $\Delta\Phi$ represents a discrete approximation to the right-hand side of Eq. (1), and Δt is the time step-size. Then, at each time step $(m+1)\Delta t$, we update the value of the level set function Φ at each grid point x ((i, j) in 2D and (i, j, k) in 3D) from its previous value Φ^m , until convergence or after a user specified number of time steps.

In this framework, updating of the level set function Φ is performed on fixed grid points; thus, no parameterization of the curves or surfaces is needed during the deformation. The parametric representation is only computed after the evolution is completed, that is, by taking the zero level set of the function Φ at the last time step. This step requires an isocontour or isosurface algorithm.

For efficiency, the *narrow-band method* can be used to update the level set function only at a small set of points in the neighborhood of the zero level set instead of at all the points of the computational domain. This scheme, however, requires recomputing the level set function after a certain amount of time steps, since the zero level set might move out of the updating region [9].

Another issue is that the speed function is meaningful only at the moving contour, i.e., the zero level set, and thus it is sometimes necessary to “extend” the speed function at the zero level set to the entire computational domain. One extension method can be found in [9] which aims to keep the level set function Φ to be a signed distance function throughout the evolution and thus eliminates the need for re-initialization.

It is well known that the topology change of the embedded contour is automatic during the evolution of the level set function Φ , which also means that the topology of the final contour is unpredictable.

2.2. Digital topology A 2D (resp. 3D) digital (i.e. binary) image $V \subset Z^2$ (resp. Z^3) is defined as a square (resp. cubic) array of lattice points. We follow the conventional definition of *n-neighborhood* and *n-connectivity*, where $n \in \{4, 8\}$ in 2D and $n \in \{6, 18, 26\}$ for a 3D image. We denote the *n-neighborhood* of a point x by $N_n(x)$, and the set comprising the neighborhood of x with x removed by $N_n^*(x)$. The set of all *n-connected* components of $X \subset V$ is denoted by $\mathcal{C}_n(X)$.

In order to avoid a connectivity paradox, different connectivities, n and \bar{n} , must be used in a binary image com-

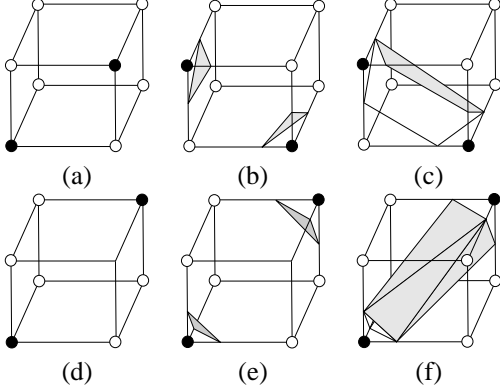


Figure 1: (a) An ambiguous face; (b) and (c) are two possible tilings. (d) An ambiguous cube; (e) and (f) are two possible tilings.

prising an object (foreground) X and a background \bar{X} . For example, in 2D, if n is chosen to be 4, then \bar{n} must be 8, and vice versa. In 3D, (18, 6) and (26, 6) are two pairs of compatible connectivities. Following [16], we distinguish the 6-connectivity associated with the 18-connectivity by 6^+ -connectivity, which is required to correctly compute the *topological numbers* (see below). The following definitions are from [16] and [18], where $M = 8$ in 2D and $M = 26$ in 3D.

Definition 1 (Geodesic Neighborhood) Let $X \subset V$ and $x \in V$. The geodesic neighborhood of x with respect to X of order k is the set $N_n^k(x, X)$ defined recursively by: $N_n^1(x, X) = N_n^*(x) \cap X$ and $N_n^k(x, X) = \cup\{N_n(y) \cap N_M^*(x) \cap X, y \in N_n^{k-1}(x, X)\}$.

Definition 2 (Topological Numbers) Let $X \subset V$ and $x \in V$. The topological numbers of the point x relative to the set X are: $T_4(x, X) = \#C_4(N_4^2(x, X))$ and $T_8(x, X) = \#C_8(N_8^1(x, X))$ in 2D; and $T_6(x, X) = \#C_6(N_6^2(x, X))$, $T_{6^+}(x, X) = \#C_6(N_6^3(x, X))$, $T_{18}(x, X) = \#C_{18}(N_{18}^2(x, X))$, and $T_{26}(x, X) = \#C_{26}(N_{26}^1(x, X))$ in 3D, where $\#$ denotes the cardinality of a set.

Topological numbers are used to classify the topology type of a grid point, especially for the characterization of *simple* points. A point is *simple* if its addition to or removal from the object does not change the topology of either the object or the background, in other words, it does not change the number of connected components, number of cavities and handles of both the foreground and the background. It is proven in [16] that a point x is *simple* if and only if $T_n(x, X) = 1$ and $T_{\bar{n}}(x, \bar{X}) = 1$, where (n, \bar{n}) is a pair of compatible connectivities.

2.3. Isosurface/isocontour algorithms In 3D, the marching cubes (MC) algorithm is a standard isosurface algorithm that produces a triangulated surface whose vertices lie on the edges of the cubic lattice [19].

As shown in Fig. 1, the way in which an isosurface intersects a cube is not always unique, which results in the so-called ambiguous face and ambiguous cube cases. The major difference between different MC algorithms lies in how they choose between the two possible tilings for the ambiguous cases. A well-accepted criterion is that the surface tiling should correctly reflect the topology of the isosurface itself. Under the assumption that the embedding function is densely sampled so that it is approximately linear on each cube, *face saddle points* and *body saddle points* can be used to produce isosurfaces that are topologically consistent with the embedded implicit surfaces [20]. We note that the saddle points are the critical points of the embedding function — that is, the points where the first order derivatives of the function are all zero.

In this paper, we need an isosurface algorithm that can correctly represent the topology of a binary object prescribed by a given digital connectivity. For this purpose, we propose the use of a *connectivity consistent* MC (CCMC) algorithm. In this algorithm, the coordinates of surface intersections are still computed through linear interpolation, but which surface tiling to choose depends on the pre-defined digital connectivity. In particular, we choose the tilings in Figs. 1(c) and 1(e) for the corresponding ambiguous cases respectively if the black points are assumed to be 18-connected while the white points are 6^+ -connected. If the black points are assumed to be 26-connected, then Figs. 1(c) and 1(f) should be used instead.

The corresponding algorithm in 2D can be called the *connectivity consistent marching squares* (CCMS) algorithm. The only ambiguous case that needs special care is an ambiguous square (e.g., the front face of the cube in Fig. 1(a)). The correct tiling should separate the white points while connect the black ones if the black points are 8-connected, and vice versa.

3. New Algorithm

Our topology preserving mechanism exploits the binary nature of the object that is delineated by the level set function. In this framework, the topology changes at the zero level set are directly related to the sign changes of the level set function. A constraint can then be imposed to keep the topology unchanged while the implicit contours deform. The resulting deformable model behaves exactly as the underlying model except at places where topology changes can occur. In particular, the model still deforms continuously, and sub-pixel accuracy is maintained.

Assume that the implicit contour is embedded as the zero level set of a level set function. Then at a given time step each grid point is either inside or outside the contour depending on the sign of the level set function at that point. We arbitrarily assign points with zero distance to be inside the contour. The topology of the digital object defined to be

inside can only change if the inside/outside status at a point is changed or, equivalently, if the level set function changes sign at a grid point. Therefore, to preserve the topology of the object, the level set function can only be allowed to change sign at simple points. This is the principle of the topology preserving constraint that we impose. We now give a detailed description of its implementation.

We follow the narrow band implementation of geometric deformable models since it is computationally fast. In the standard implementation, the level set function Φ is updated at each iteration using Eq. (3) in the narrow band, and is periodically recomputed over the whole grid after several or many iterations. In our topology-preserving implementation, it is convenient to store a binary-valued indicator function B , defined on the digital grid, which is 1 where Φ is negative or zero and is 0 otherwise. Our method replaces the computation of Eq. (3) with the following algorithm, where the level set function Φ at time $m + 1$ is to be updated from its previous value Φ^m :

Algorithm 1 (Topology Preserving Update) For each grid point x in the narrow-band:

1. Compute $\Phi_{\text{temp}}(x) = \Phi^m(x) + \Delta t \Delta \Phi^m(x)$.
2. If $\Phi_{\text{temp}}(x)$ has the same sign as $\Phi^m(x)$, then set $\Phi^{m+1}(x) = \Phi_{\text{temp}}(x)$ and go to Step 6. Otherwise continue to Step 3.
3. Compute the topological numbers $T_n(x, X)$ and $T_{\bar{n}}(x, \bar{X})$, where (n, \bar{n}) is the chosen digital connectivity pair, $X = \{x | B(x) = 1\}$, and $\bar{X} = \{x | B(x) = 0\}$.
4. If the point is simple — i.e., $T_n(x, X) = T_{\bar{n}}(x, \bar{X}) = 1$ — then set $\Phi^{m+1}(x) = \Phi_{\text{temp}}(x)$, $B(x) = (B(x) + 1) \bmod 2$, and go to Step 6. Otherwise continue to Step 5.
5. Point x is not simple. Do not allow the front to pass over x by making sure Φ^m does not change sign at x . Accordingly, set $\Phi^{m+1}(x) = \epsilon \Phi^m(x)$, where ϵ is a small positive number.
6. Pick up the next point x in the narrow band, and go to Step 1. ■

We note that there can be some arbitrariness in the specific result of this algorithm depending on the order in which the points are visited in the narrow band. This situation is well known in skeletonizing algorithms where the result depends on the order of simple point removal. Currently, we do not have a criterion to prefer one ordering over another one. But we feel that this problem is not as significant as in skeletonizing since the overall motion of the deformable model is controlled by its speed terms. The simple point criterion only takes place at locations where topological changes are bound to occur otherwise, which is ordinarily a very small portion of the overall deforming contour. In the experiments reported in the paper, we followed exactly the same ordering as in the standard narrow band

implementation, where points are ordered by their natural coordinates.

To accelerate the convergence rate, most geometric deformable models are implemented in a multiscale fashion [5]. The level set function is first constructed and evolved on a coarser grid than the underlying image, then it is upsampled to a finer resolution after it converges in the coarser scale. Usually, a first-order, linear interpolation is applied to refine the level set function. This procedure, however, may yield a different topology than the coarse volume. To be consistent with our approach, the correct upsampling method is zero-order duplication, which copies the value of a point at the coarser grid to all its children at the finer grid. This preserves the topology from coarse to fine scales.

After the level set iterations have converged, we extract the final contour using the CCMC (or CCMS) algorithm. In these algorithms, the surface or contour location is computed by linear interpolation of the level set function, but the tiling for the ambiguous cases is selected based on the chosen digital connectivity pair. If the level set function value is exactly zero at a grid point, it is explicitly adjusted before interpolation to prevent a singularity in the resulting surface mesh or contour.¹ Since we consider zero-valued points to be inside points, i.e., as negative distance points, we set a zero function value to some small negative value, say $-\epsilon$.

4. Experiments

In this section, we present several experiments that apply our new topology preserving geometric deformable models in 2D and 3D. Since the new models can be obtained by imposing the topology preservation constraint on existing geometric deformable models (GDMs), we will refer to the *standard* models without topology constraint as SGDMs and the corresponding (i.e., with the same set of speed terms) *topology* preserving models as TGDMs. Note that in TGDMs, the CCMC or CCMS algorithm must be used in order to correctly extract the final curves or surfaces from the level set function, while the SGDMs require a standard isocontour or isosurface algorithm (preferably using face and body saddle points). In the following experiments, we choose $(n, \bar{n}) = (4, 8)$ as the pair of 2D digital connectivities and $(n, \bar{n}) = (18, 6^+)$ for 3D.

Fig. 2 shows a 2D example that illustrates the topology preservation ability of a TGDM model. Fig. 2(a) shows the original image (65×140 pixels) consisting of two circular cells placed side-by-side. The two initial curves are also shown on this figure. Figs. 2(b) and 2(c) show the location of the implicit curves of the SGDM at an intermediate and the final stage. Because of the weak edge between the two cells, the two initial curves merged into one final curve — an undesirable result in this example. Figs. 2(d)

¹This is one of several major artifacts that exist in most existing isosurface and isocontour softwares.

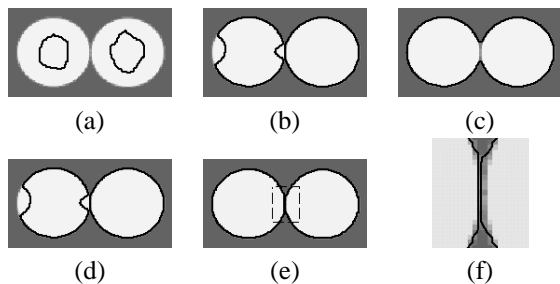


Figure 2: A 2D image and boundary detection using both SGDM and TGDM (see text for details).

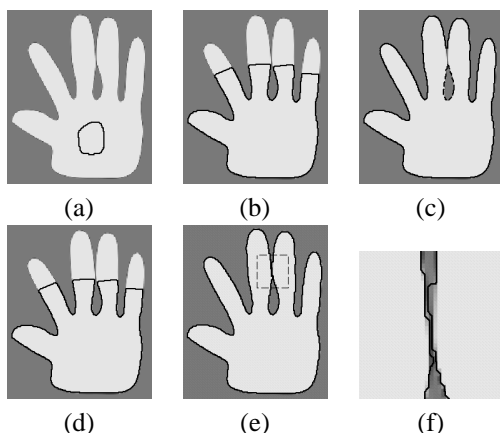


Figure 3: Segmentation of a hand phantom using both SGDM and TGDM (see text for details).

and 2(e) show the corresponding deformations when the topology preservation constraint is enforced (and otherwise the two models are identical in internal and external force terms). We note that TGDM keeps the two curves separated throughout the evolution and also correctly finds the boundary of each cell. Fig. 2(f) demonstrates the sub-pixel resolution of the result.

Fig. 3 shows another 2D example in which the same models as in the previous example were applied to find the boundary of a hand-shaped object. The original image (220×190 pixels) and the initial curve are shown in Fig. 3(a). Figs. 3(b) and 3(c) illustrate the deformation of the SGDM contour at an intermediate and the final stage. Again, without the topology preservation constraint, the initial curve changes topology and gives two separate curves as the final result (the dark outer curve and the dashed inner curve of Fig. 3(c)). We note that the two middle fingers in this hand become one with a hole in it in the final segmentation. The corresponding deformations of the TGDM contour are illustrated in Figs. 3(d) and 3(e). TGDM keeps the boundary of each finger separated, and the final contour correctly reflects the shape of the hand, as can be seen clearly in the zoomed view of Fig. 3(f).

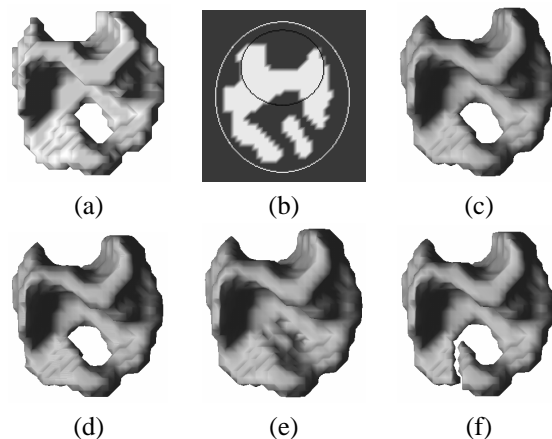


Figure 4: A 3D phantom and the segmentation results of using both SGDM and TGDM and two different initializations.

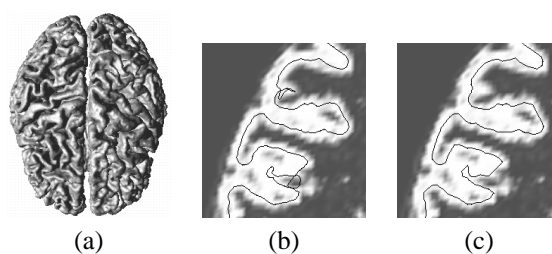


Figure 5: (a) Result of cortical surface reconstruction. (b) Self-intersection from PDM, and (c) no intersection with TGDM.

Next, we applied a 3D version of the geometric deformable model of Eq. (2) to find the boundary surface of the 3D object depicted in Fig. 4(a). The object is actually a piece of a white matter (WM) volume segmentation from a magnetic resonance brain image. Due to data noise, the WM piece has a handle in it, which is the wrong topology from an anatomical standpoint. In fact, we desire a topology equivalent to that of a sphere. We applied both SGDM and TGDM starting from two different initializations: a large sphere that encloses the whole object and a small ellipsoid that intersects with the object. A 2D slice showing the object and the two initial surfaces is shown in Fig. 4(b).

Figs. 4(c) and 4(d) are the final surfaces obtained by SGDM. The two results are the same since geometric deformable models are insensitive to initialization. The final surfaces both have a handle, however, which is the incorrect topology. With the sphere as the initialization, TGDM gives the final surface shown in Fig. 4(e), and with the ellipsoid, it gives the result shown in Fig. 4(f). Both surfaces have the correct topology, but the topology is preserved in a different way. The surface obtained from the sphere initialization yields a thin membrane across the tunnel through the original object, while the ellipsoid initialization makes a cut in the handle.

Our final experiment applies SGDM, TGDM, and a parametric deformable surface model to extract the central cortical surface from an initial fuzzy segmentation of a brain MRI image volume. We used exactly the same initialization, the same external forces and similar internal forces for the geometric deformable models as in the parametric one. The results are presented in Fig. 5.

Fig. 5(a) shows the final surface extracted from the parametric model. The SGDM and TGDM surfaces look very similar, but on close examination there are important differences. The parametric model result, for example, has self-intersections as shown in Fig. 5(b), while the TGDM surface does not, as shown in Fig. 5(c). Also, the genus (number of handles) of the SGDM result is 40, while that of both the parametric model result and the TGDM result is 0. Thus, TGDM produces both the correct topology and a valid manifold; hence it is the only model that gives a legal cortical surface reconstruction.

5. Discussion and Conclusion

The example shown in Fig. 4 points out a weakness in our overall approach that should be addressed in future work. First, the result can clearly depend on the initialization in a dramatic way. The two results, one that fills the tunnel and the other that breaks the handle, are dramatically different ways to address the topological preservation. At present we have no formulation of an optimality criterion that would choose one of these solutions over the other. This is not atypical in deformable models, where the particular initialization very often determines the exact details of the final solution, but it deserves to be addressed nonetheless.

We note that the topological numbers are computed locally, which makes the simple point checking process straightforward and efficient. As a result, the topology constraint does not add much computational burden as compared to the standard geometric deformable models. For the phantom experiments, the time difference between standard and new geometric models is barely noticeable; and for the brain cortical surface reconstruction, the extra time taken by the topology constraint enforcement is less than 7 percent of the total processing time.

In summary, we have developed a class of new geometric deformable models where the topology of the implicit curves or surfaces is preserved throughout the deformation. The topology is preserved by checking a simple point criterion during the level set evolution, which requires a relatively straightforward modification to standard implementation of geometric deformable models. Experiments were conducted to show the success of the new models and illustrate their potential applications.

Acknowledgments This work was supported in part by NSF/ERC grant CISST#9731748 and NIH/NINDS grant R01NS37747.

References

- [1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comp. Vision*, 1:312–333, 1988.
- [2] L. D. Cohen, "On active contour models and balloons," *CVGIP: Image Understanding*, 53:211–218, 1991.
- [3] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Trans. Imag. Proc.*, 7(3):359–369, 1998.
- [4] V. Caselles, F. Catte, T. Coll, and F. Dibos, "A geometric model for active contours in image processing," *Numerische Mathematik*, 66:1–31, 1993.
- [5] R. Malladi, J. A. Sethian, and B. C. Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE Trans. PAMI*, 17:158–175, 1995.
- [6] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *Int J. Computer Vision*, 22:61–79, 1997.
- [7] A. Yezzi, S. Kichenassamy, P. Olver, and A. Tannenbaum, "A geometric snake models for segmentation of medical imagery," *IEEE Trans. Med. Imaging*, 16:199–209, 1997.
- [8] K. Siddiqi, Y. B. Lauziere, A. Tannenbaum, and S. W. Zucker, "Area and length minimizing flow for shape segmentation," *IEEE Trans. Image Proc.*, 7:433–443, 1998.
- [9] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge, UK: Cambridge University Press, 2 ed., 1999.
- [10] T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: A survey," *Medical Image Analysis*, 1(2):91–108, 1996.
- [11] C. Xu, D. L. Pham, M. E. Rettmann, D. N. Yu, and J. L. Prince, "Reconstruction of the human cerebral cortex from magnetic resonance images," *IEEE Trans. Med. Imaging*, 18(6):467–480, 1999.
- [12] D. MacDonald, N. Kabani, D. Avis, and A. C. Evans, "Automated 3-D extraction of inner and outer surfaces of cerebral cortex from MRI," *NeuroImage*, 12:340–356, 2000.
- [13] D. W. Shattuck and R. M. Leahy, "Topologically constrained cortical surfaces from MRI," in *Proceedings of the SPIE*, vol. 3979, (San Diego, USA), pp. 747–758, February 2000.
- [14] B. Fischl, A. Liu, and A. M. Dale, "Automated manifold surgery: Constructing geometrically accurate and topologically correct models of the human cerebral cortex," *IEEE Trans. Med. Imaging*, 20(1):70–80, 2001.
- [15] P. K. Saha and B. B. Chaudhuri, "Detection of 3D simple points for topology preserving transformations with application to thinning," *IEEE Trans. PAMI*, 16:1028–1032, 1994.
- [16] G. Bertrand, "Simple points, topological numbers and geodesic neighborhoods in cubic grids," *Pattern Recognition Letters*, 15:1003–1011, 1994.
- [17] J.-F. Mangin, V. Frouin, I. Bloch, J. Regis, and J. Lopez-Krahe, "From 3D magnetic resonance images to structural representations of the cortex topography using topology preserving deformations," *J. Math. Imag. Vision*, 5:297–318, 1995.
- [18] G. Bertrand, J. C. Everat, and M. Couprie, "Image segmentation through operators based on topology," *Journal of Electronic Imaging*, 6:395–405, 1997.
- [19] W. E. Lorensen and H. E. Cline, "Marching cubes: A high-resolution 3D surface construction algorithm," *ACM Computer Graphics*, 21(4):163–170, 1987.
- [20] B. K. Natarajan, "On generating topologically consistent iso-surfaces from uniform samples," *Visual Computer*, 11(1):52–62, 1994.