A Topology Preserving Level Set Method for Geometric Deformable Models

Xiao Han, *Student Member*, *IEEE*, Chenyang Xu, *Member*, *IEEE*, and Jerry L. Prince, *Senior Member*, *IEEE*

Abstract—Active contour and surface models, also known as deformable models, are powerful image segmentation techniques. Geometric deformable models implemented using level set methods have advantages over parametric models 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 object to be segmented has a known topology that must be preserved. In this paper, we present a new class of geometric deformable models designed using a novel topology-preserving level set method, which achieves topology preservation by applying the simple point concept from digital topology. These new models maintain the other advantages of standard geometric deformable models including subpixel accuracy and production of nonintersecting curves or surfaces. Moreover, since the topology-preserving constraint is enforced efficiently through local computations, the resulting algorithm incurs only nominal computational overhead over standard geometric deformable models. Several experiments on simulated and real data are provided to demonstrate the performance of this new deformable model algorithm.

Index Terms—Geometric deformable model, topology preservation, topological constraint, level set method, digital topology, simple points, active contours.

1 Introduction

Deformable models are object-delineating curves or surfaces that move within two-dimensional (2D) or three-dimensional (3D) digital images under the influence of both internal and external forces and user defined constraints. 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 broadly classified as either *parametric deformable models* (see [1], [2], [3], [4], [5]) or *geometric deformable models* (see [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]) according to their representation and implementation. In particular, parametric deformable models are represented *explicitly* as parameterized contours[1] (i.e., curves or surfaces) in a Lagrangian framework. They are the older of the two formulations and have been extensively used in many applications (see [17], for example). Geometric deformable models, on the other hand, are represented *implicitly* as level sets of higher-dimensional, scalar level set functions and evolve in an Eulerian fashion [18]. Geometric deformable models were introduced more recently by Caselles et al. [6] and by Malladi et al. [7].

Geometric deformable models have several important advantages over parametric models. First, they are completely intrinsic and, therefore, are independent of the parameterization of the evolving contour. In fact, the model is generally not parameterized until evolution of the level set function is complete. Thus, there is no need to add or remove nodes from an initial parameterization or adjust the spacing of the nodes as in parametric models. Second, the intrinsic geometric properties of the contour, such as the unit normal vector and the curvature, can be easily computed from the level set function. This contrasts with the parametric case, where inaccuracies in the calculations of normals and curvature result from the discrete nature of the contour parameterization. Third, the propagating contour can automatically change topology in geometric models (e.g., merge or split) without requiring an elaborate mechanism to handle such changes as in parametric models (see [19], [20]). Finally, the resulting contours do not contain self-intersections, which are computationally costly to prevent in parametric deformable models (see [21]).

Topological flexibility has long been claimed as a major advantage of geometric deformable models over parametric deformable models. Such flexibility is so desirable in some applications that methods to adaptively change the contour topology have also been developed for parametric deformable models [19], [20]. But, topological flexibility is not always desired. In particular, when a specific object (target) is sought and its composition—i.e., the number of components and the homology of each component—is known, then it is most natural to seek the target in a way that yields the correct composition or topology. For example, in the analysis of 3D brain images—the application that motivated our work on this subject—it is desirable that a reconstruction of the cortical

1. In this paper, we use the word *contour* to refer to either a curve or surface, and the words curve and surface are used explicitly only when the dimensionality must be clear.

- *X. Han and J.L. Prince are with the Electrical and Computer Engineering Department, Johns Hopkins University, 105 Barton Hall, 3400 North Charles Street, Baltimore, MD 21218. E-mail: {xhan, prince}@jhu.edu.*
- *C. Xu is with the Imaging and Visualization Department, Siemens Corporate Research, Inc., 755 College Road East, Princeton, NJ 08501. E-mail: chenyang.xu@scr.siemens.com.*

Manuscript received 13 May 2002; revised 14 Nov. 2002; accepted 20 Nov. 2002.
Recommended for acceptance by Y. Amit.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 116535.

surface have a topology that is consistent with brain anatomy [22], [21]. Recently, in fact, there have been several post-processing methods reported to correct the topology of a cortical segmentation that has the wrong topology [23], [24], [25]. In this application, and others like it, the topology flexibility of geometric deformable models is considered to be a liability rather than an advantage [25].

When topology preservation is desired, parametric deformable models are typically used because topology is explicitly maintained by their Lagrangian formulation. Self-intersections can become a problem in these algorithms, however, when external forces drive the model vertices together and step sizes are simultaneously selected to be large in order to reduce convergence time. When simple contours are required—as is usually the case in image segmentation—some additional computations are necessary in order to avoid self-intersections. Unfortunately, the computational demands related to self-intersection detection are very high, especially for surfaces [21], and most parametric models neglect this step, relying on smooth external forces and extremely small step sizes instead. Geometric deformable models inherently prevent self-intersections because of the level set representation as well as the entropy conditions imposed during level set evolution [26], [18] and the way isocontours are typically computed (see [27]). But, prior to this paper, there has been no way to take advantage of this property (and the other nice properties of geometric deformable models) *and* to prevent topological changes during the level set evolution. The difficulty of developing a level set method that preserves topology has been noted by Hermosillo et al. [28]. They speak of the need for a topology-preserving evolution and then comment "[f]or planar curves, such an evolution is given by the curvature flow, but unfortunately this is not the case for surfaces. Much research has been devoted to this problem, but it remains an open one."

In this paper, we develop a *topology-preserving level set method* (TLSM) for geometric deformable models that guarantees that the final contour has exactly the same topology as the initial one and does not contain any self-intersection. 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 [29], [30], [31]. We note that our approach maintains the subpixel interpolation and boundary regularization properties of geometric deformable models, which distinguishes our method from the topology-preserving region growing method of Mangin et al. [32]. The TLSM 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 deformable models, which we will refer to as *topology-preserving geometric deformable models* (TGDM's).[2]

The remainder of the paper is organized as follows: In Section 2, we present the basic notation and key ideas of the geometric deformable models. We then present our topology-preserving framework in Section 3. Experimental results on both 2D and 3D phantoms and real data are shown in Section 4 to demonstrate the behavior and advantages of the new TGDM's, which also serve as an illustration for their potential applications. Section 5 summarizes the method, discusses the results, and gives more details on the connections between previous work and our approach. Finally, we give a brief conclusion in Section 6.

We note that preliminary results related to this work have been described in a conference paper [34] and its application to brain cortex segmentation has been described in [35], [36].

2 Geometric Deformable Models

Geometric deformable models are based on the theory of front evolution and are implemented using the level set numerical method [18]. In this section, we briefly review the main theory and major results of geometric deformable models.

2.1 Front Evolution and Level Set Theory

Let $C(\mathbf{p},t)$, defined as $\{x(\mathbf{p},t), y(\mathbf{p},t)\}$ in 2D and $\{x(\mathbf{p},t), y(\mathbf{p},t), z(\mathbf{p},t)\}$ in 3D, denote a family of closed contours (i.e., curves or surfaces) generated by evolving an initial contour $C_0(\mathbf{p}) = C(\mathbf{p},0)$, where t parameterizes the family and $\mathbf{p}$ parameterizes the given contour. The basic result from the front evolution theory is that the geometric shape of the contour is determined by the normal component of the evolution velocity, while the tangential component affects only the parameterization. Hence, after a possible reparameterization, the evolution equation can be written as

$$\begin{cases} \frac{\partial C(\mathbf{p},t)}{\partial t} = F(C(\mathbf{p},t))\vec{n}(C(\mathbf{p},t)), \\ C(\mathbf{p},0) = C_0(\mathbf{p}), \end{cases} \tag{1}$$

where $F(C(\mathbf{p},t))$ is a scalar function that often depends on the curvature κ of the contour (for surfaces, both mean and Gaussian curvatures can be used), and $\vec{n}(C(\mathbf{p},t))$ is the unit normal vector (conventionally chosen to be the inward normal) along the contour $C(\mathbf{p},t)$.

The Lagrangian approach to the above evolution equation involves discretizing the contour into a set of elements (e.g., nodes connected by lines or triangles) and updating the node positions using a numerical approximation to (1). This is the approach of parametric deformable models. Frequent adjustment of the node spacing is required in order to preserve data fidelity and reduce numerical approximation errors. Computationally complex approaches may also be required for self-intersection avoidance.

The level set technique developed by Osher and Sethian [26] represents the contour $C(\mathbf{p},t)$ implicitly as the zero level set of a smooth, Lipschitz-continuous scalar function $\Phi(\mathbf{x},t)$, also known as the *level set function*, where $\mathbf{x} \in \mathcal{R}^2$ in 2D and $\mathbf{x} \in \mathcal{R}^3$ in 3D. The implicit contour at any time t is given by $C(\cdot,t) = \{\mathbf{x} | \Phi(\mathbf{x},t) = 0\}$. Although there are infinite many choices of the level set function, in practice, the signed distance function is preferred for its stability in numerical computations. The fast marching method proposed in [37], [38] provides an efficient algorithm for constructing the signed distance function from a given initial contour. We used signed distance functions constructed in this way for all of our experiments in this paper.

By differentiating $\Phi(\mathbf{x},t) = 0$ with respect to t and substituting (1), the following associated equation of motion for the level set function $\Phi(\mathbf{x},t)$ can be derived:

$$\begin{cases} \frac{\partial \Phi(\mathbf{x},t)}{\partial t} = F(\mathbf{x},t)|\nabla \Phi(\mathbf{x},t)|, \\ \Phi(C_0(\mathbf{p}),0) = 0, \end{cases} \tag{2}$$

2. The GDM acronym used here does not mean *geometrically deformable models*, which is a different concept introduced by J. Miller et al. [33] and shares the same acronym.

where ∇ is the gradient operator and $|\nabla\Phi|$ denotes the norm of the gradient of Φ. Note that the function $F(\mathbf{x},t)$ is only defined at the contour location originally and, hence, needs to be extended to the whole computational domain (see [18]) in order that (2) applies to the whole space.

2.2 Geometric Deformable Models

Caselles et al. [6] and Malladi et al. [7] applied the above theory to the problem of image segmentation by multiplying the contour velocity by a "stopping" term $g(|\nabla I(\mathbf{x})|)$ that is a monotonically decreasing function of the gradient magnitude of the image I (or its smoothed version). In this way, they arrived at the following evolution equation

$$\frac{\partial\Phi(\mathbf{x},t)}{\partial t} = g(|\nabla I(\mathbf{x})|)(c+\kappa(\mathbf{x},t))|\nabla\Phi(\mathbf{x},t)|, \tag{3}$$

where c is a constant inflation or deflation (depending on its sign) speed term that aims to keep the contour moving in the proper direction, and $\kappa(\mathbf{x},t)$ is the (mean) curvature of the level set of $\Phi(\cdot,t)$ that passes through the point $\mathbf{x}$, which can be easily computed from the spatial derivatives of $\Phi(\cdot,t)$ (see [18]). We note that, in [6], [7], the above formulation is originally derived for planar curves, however, the very same form applies to surfaces as well. In the remainder of this paper, all of the equations apply to both curves and surfaces unless stated otherwise.

The model of (3) does not arise from the minimization of an energy functional as in the classical active contour models. To address this, Caselles et al. [8], [9] and Kichenassamy et al. [10], [11] derived another geometric deformable model, called the *geodesic active contour model*. The basic idea is to consider the object boundary detection as a problem of geodesic computation in a Riemannian space, according to a metric $g(\mathbf{x})$ induced by the given image I. This idea can be formally written as

$$\min_C J(C) = \int g(C(\mathbf{p}))dC, \tag{4}$$

where dC denotes the arc-length in 2D or the infinitesimal area element in 3D, and $g(\mathbf{x})$ is usually chosen to be the same as the stopping term $g(|\nabla I(\mathbf{x})|)$ used in the previous model.

Minimizing $J(C)$ using a steepest descent algorithm starting from an initial contour C_0 gives the following contour evolution equation

$$\begin{cases}\frac{\partial C(\mathbf{p},t)}{\partial t} = (g(C(\mathbf{p},t))\kappa(C(\mathbf{p},t)) - \nabla g(C(\mathbf{p},t))\cdot\vec{n}(C(\mathbf{p},t)))\vec{n}(C(\mathbf{p},t)),\\ C(\mathbf{p},0) = C_0(\mathbf{p}).\end{cases} \tag{5}$$

This geodesic active contour model can be readily cast within the level set framework. This yields an equivalent contour evolution process implemented using the following level set function evolution equation

$$\begin{aligned}\frac{\partial\Phi(\mathbf{x},t)}{\partial t} &= g(\mathbf{x})|\nabla\Phi(\mathbf{x},t)|\mathrm{div}\left(\frac{\nabla\Phi(\mathbf{x},t)}{|\nabla\Phi(\mathbf{x},t)|}\right) + \nabla g(\mathbf{x})\cdot\nabla\Phi(\mathbf{x},t)\\ &= g(\mathbf{x})\kappa(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)| + \nabla g(\mathbf{x})\cdot\nabla\Phi(\mathbf{x},t),\end{aligned} \tag{6}$$

where $\mathrm{div}(\cdot)$ denotes the divergence of its argument, and $\kappa(\mathbf{x},t)$ is the (mean) curvature as in (3).

There are many other extensions of the basic geometric deformable model in the literature (e.g., [13], [39], [40], [14], [41], [15], [16]) which were designed either to improve the overall performance of the original model or to adapt to particular applications. In this work, we consider a very general framework summarized by the following evolution equation [18], [41]:

$$\begin{aligned}\frac{\partial\Phi(\mathbf{x},t)}{\partial t} &= F_{\mathrm{prop}}(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)| + F_{\mathrm{curv}}(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)|\\ &\quad + \vec{F}_{\mathrm{adv}}(\mathbf{x},t)\cdot\nabla\Phi(\mathbf{x},t),\end{aligned} \tag{7}$$

where $F_{\mathrm{prop}}(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)|$ is an expansion or contraction force or speed (people use "force" and "speed" interchangeably); $F_{\mathrm{curv}}(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)|$ is the part of the force that depends on the intrinsic geometry, especially the (mean) curvature $\kappa(\mathbf{x},t)$; and $\vec{F}_{\mathrm{adv}}(\mathbf{x},t)\cdot\nabla\Phi(\mathbf{x},t)$ is an advection force that passively transports the contour.

The right-hand side of (7) can arise from the gradient descent minimization of an energy functional as in the geodesic active contour model (6), where $F_{\mathrm{prop}}(\mathbf{x},t)=0$, $F_{\mathrm{curv}}(\mathbf{x},t)=\kappa(\mathbf{x},t)g(\mathbf{x})$, and $\vec{F}_{\mathrm{adv}}(\mathbf{x},t)=\nabla g(\mathbf{x})$. In general, however, one can choose a different form for each force term for a given purpose. As an example, we can choose $F_{\mathrm{prop}}(\mathbf{x},t)=R(\mathbf{x})$ to be a region force[3] (see [15], [41]) or a binary flow force [14], $F_{\mathrm{curv}}(\mathbf{x},t)$ to be proportional to the (mean) curvature $\kappa(\mathbf{x},t)$, and $\vec{F}_{\mathrm{adv}}(\mathbf{x},t)=\vec{\mathbf{v}}(\mathbf{x})$ to be a gradient vector flow force [4]. With these choices the evolution equation becomes

$$\begin{aligned}\frac{\partial\Phi(\mathbf{x},t)}{\partial t} &= \omega_R R(\mathbf{x})|\nabla\Phi(\mathbf{x},t)| + \omega_\kappa\kappa(\mathbf{x},t)|\nabla\Phi(\mathbf{x},t)|\\ &\quad + \omega_{\vec{\mathbf{v}}}\vec{\mathbf{v}}(\mathbf{x})\cdot\nabla\Phi(\mathbf{x},t),\end{aligned} \tag{8}$$

where ω_R, ω_κ, and $\omega_{\vec{\mathbf{v}}}$ are weights for the respective forces. For a binary-valued image I having values zero or one, it is convenient to define $R(\mathbf{x})=2I(\mathbf{x})-1$ to provide an expansion force inside the object and a contraction force outside. The model in (8) is used in the 3D experiments presented later in this paper.

2.3 Numerical Implementation

One advantage of the geometric deformable model is that, even though the implicit contour itself can develop singularities (like cusps and corners) and can merge or split to change topology, the level set function Φ remains well-defined. Thus, one can discretize the level set evolution equation on a fixed Cartesian grid and use a finite difference scheme to robustly solve the evolution equation numerically. In order to capture the singularities that might develop along the implicit contour, Osher and Sethian [26] proposed an upwind scheme that incorporates piecewise continuous approximations to Φ and utilizes one-sided (or upwind) derivatives in the approximation of $\nabla\Phi$. The scheme is numerically stable and produces an entropy-satisfying viscosity solution to (7).

Denote a grid point by $\mathbf{x}_i$ and the discrete time scale by t_m, where i,m are integers. The resulting level set update equation can be written as

$$\Phi(\mathbf{x}_i,t_{m+1}) = \Phi(\mathbf{x}_i,t_m) + \Delta t\Delta\Phi(\mathbf{x}_i,t_m), \tag{9}$$

where $\Delta t = t_{m+1}-t_m$ is the time-step size. Since we are interested in a generic geometric deformable model, we use $\Delta\Phi$ to denote the upwind finite difference approximation to

3. Also known as a signed pressure force.

the right-hand side of (7) (see [18] for an explicit formula). Given an initial level set function $\Phi(\cdot, t_0)$, (9) can be used to update the level set function at successive time instants $t_{m+1}, m = 0, 1, \ldots,$ until convergence. Although not explicitly computed until the end, the zero level set of $\Phi(\cdot, t_m)$, $m = 1, 2, \ldots$ represents the evolving contour(s).

As mentioned before, the forces are really only meaningful at the moving contour itself, i.e., the zero level set of Φ. Yet, the update equation (9) applies to all values of Φ, not just those around zero. In fact, it is clear from (6) that, in this implementation of the geodesic deformable model, the forces have been "naturally" extended to apply to all level sets, not just the zero level set. By "naturally," it is meant that the same expression is used to evaluate the forces over the whole computational domain. One implication of this particular force extension is that all level sets are attracted to the desired image feature, which tends to crowd the level sets closer together as the iterations proceed. Because of this, periodic reinitialization of the level set function (using the fast marching method, for example) is required in order that it closely approximates a signed distance function; this improves numerical stability and accuracy of the overall computation. An alternate extension method that preserves Φ at any time as a signed distance function was presented in [42], [18], but this requires more computation per iteration and is generally much slower than this simple periodic reinitialization scheme.

There are several ways to increase the computational speed of geometric deformable models including time-implicit numerical schemes and the narrow band method. In time-implicit numerical schemes [43], [44], the level set function at the current time step is updated from its previous values by solving a system of linear equations, which means that the level set function at the grid points are updated all at once. Time-implicit schemes, however, are not compatible with the topology-preserving mechanism that we describe herein, since they do not permit points to be controlled individually. We require a time-explicit step in order to be able to maintain explicit control of topology at each iteration. The narrow band method [45], [46] is perfectly compatible with our methods and, in fact, provides a considerable computational advantage since only a small set of grid points near the zero level set are modified during each iteration. Furthermore, our method can be expressed as a small, but critically important modification to the standard narrow band method. For this reason, we now give the explicit steps of the narrow band implementation of a geometric deformable model.

Algorithm 1: Narrow Band Algorithm

1. *Initialize*—Set $m = 0$ and $t_0 = 0$. Initialize $\Phi(\cdot, 0)$ to be the signed distance function of the initial contour.
2. *Build the Narrow Band*—Find the narrow band points. These are the grid points x_i whose distance $|\Phi(x_i, t_m)|$ is less than the specified narrow band width.
3. *Update*—Set $t_{m+1} = t_m + \Delta t$. For every narrow band point x_i, update its level set function value $\Phi(x_i, t_{m+1})$ using (9).
4. *Reinitialize*—If necessary, reinitialize $\Phi(\cdot, t_{m+1})$ to be the signed distance function of its own zero level set.
5. *Convergence Test*—Check whether the iterations have converged. If yes, stop; otherwise set $m = m + 1$. If reinitialization was performed in Step 4, then go to Step 2 to rebuild the narrow band; otherwise, go to Step 3.

It is worth making a few comments about the narrow band method. First, we note that in Step 3, the narrow band points can be processed in an arbitrary order since each point is updated using function values from the previous time-step. Second, reinitialization of the level set function is periodically required not only to prevent "bunching" as described above, but also to prevent the zero level set from moving out of the current narrow band (see [18]). Third, the topology of the embedded contour is normally free to change in an arbitrary fashion during the evolution of Φ. This means that the topology of the final contour is ordinarily unpredictable; images with clutter or noise can very easily produce unexpected topological results involving multiple objects, nested objects, or handles (which are found only on surfaces).

3 Topology-Preserving Level Set Method

In this section, we describe a mechanism to preserve the topology of one or multiple implicit contours during the evolution of the embedding level set function. We start with an overview of the basic principles underlying this work, especially the *digital embedding* of the implicit contour topology. We then review the fundamental concepts and notation from digital topology and introduce the definition and computation of "simple" points. We also present our topology-preserving narrow band algorithm. To better understand the convergence properties of this algorithm, we then present an interpretation of this algorithm as a constrained gradient descent algorithm in the special case of the geodesic deformable model. We then introduce 2D and 3D connectivity consistent isocontour algorithms that are guaranteed to produce topologically correct explicit representation of the implicit contour embedded in a level set function.

3.1 Overview of Basic Principles

Digital Embedding of Topology. Although geometric deformable models are formulated on the continuum, in practice they are always implemented on a digital domain —i.e., on a lattice of grid points connected by grid cells or *voxels*. Without restrictions on their functional form, there are, in general, an infinite number of contours having the same sampled level set function. Since these contours can have different numbers of components with different topologies, it is clear that it is generally impossible to recover the "true" topology of an arbitrary implicit contour from samples of its level set function. Therefore, in order to give meaning to the idea of "preserving topology" in a geometric deformable model, we must adopt certain conventions about the nature of the implied contour given its sampled level set function. The convention we describe below addresses the following two broad ambiguities. First, a continuous implicit contour might be entirely contained in one voxel or it might intersect a voxel boundary any number of times. These phenomena basically describe types of high frequency behavior not captured by the digital samples. Second, even if the contour is slowly varying, there might still be ambiguities as to how a cell is actually partitioned by a contour (see Section 3.4 and the figures therein). This ambiguity is directly tied to the classical problems of ambiguous voxels and faces in isocontour algorithms.

To resolve these topological ambiguities, in this paper, we adopt a digital interpretation of the implicit contour topology. First, we assume that the zero level set changes sufficiently slowly that it can only pass between neighboring grid points once at most. In adopting this assumption,

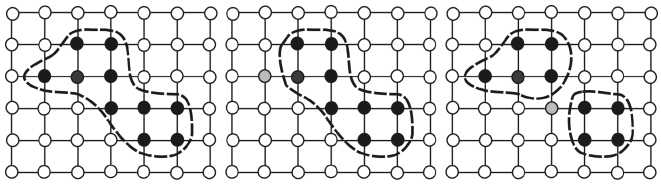


Fig. 1. Topology equivalence of the embedded contour and the digital object it defines on the discrete grid: four-connectivity for dark points and eight-connectivity for others. (a) Original contour. (b) The contour passes over a simple point. (c) The contour splits at a nonsimple point.

we are thereby ignoring topological details of the zero level set that cannot be recovered under a given discretization of the computational domain. As shown in Fig. 1, this assumption ties the topology of the zero level set with that of the digital object it encircles. More specifically, we classify grid points for which $\Phi < 0$ as *inside* the zero level set, and for which $\Phi > 0$ as *outside*. Then, the digital object consists of all the *inside* points. To avoid further ambiguity, we also adopt the convention that grid points for which $\Phi = 0$ are considered to be inside the zero level set.

The second ambiguity is resolved by specifying a pair of consistent connectivity rules for the digital object (i.e., the foreground) and its background. For example, in 2D, we might choose the object to be four-connected, in which case the background must be eight-connected (see [29] for the definition of digital connectivities in both 2D and 3D). Alternatively, we could choose the foreground to be eight-connected and the background to be four-connected. The consistent connectivity rules in 3D are $(6, 18)$, $(6, 26)$, $(18, 6)$, and $(26, 6)$, where the first number in each pair is the foreground connectivity and the second number is the background connectivity. These rules prevent topological anomalies that might, for example, allow a closed path in the background to pass through a connected foreground component.

From now on, we always treat the topology of the zero level set to be equivalent to the topology of the boundary of the digital object it defines. We refer to this as the *digital embedding* of the zero level set topology.

Topology Preservation. The digital embedding also simplifies the topology preservation problem. Since the digital object is defined by thresholding the level set function at the zero isovalue, it is clear that the topology of the implicit contour can change only if the level set function changes sign at a grid point,[4] which corresponds to a point moving from inside the digital object to the background or vice versa.

From the above discussion, we conclude that it is only necessary to be concerned about topological changes when the level set function is going to change sign. But, switching a grid point from background to foreground (or vice versa) does not necessarily change the object's topology. In fact, from the theory of digital topology (see review in the next section), we find that the topology of the digital object will *not* change if the grid point under consideration is a so-called *simple point* [29], [30], [31], [47], as illustrated in Fig. 1b. On the other hand, if the grid point is not a simple point, as illustrated in Fig. 1c, then the digital object's topology *will* change. Now, our entire strategy becomes clear. During the evolution of the level set function, we monitor the level set function for potential sign changes. If the sign is scheduled to change at a simple point, then it is allowed, but sign changes at nonsimple points are not allowed. This prevents topology changes of the underlying digital object and of the implicit zero level set as well. We note that the deforming implicit contour need not "get stuck" at a nonsimple point, since the point can become simple after additional evolution of the contour; several examples of this type of behavior are shown in Section 4.

There are two key observations to make about this overall approach. First, since it is necessary to explicitly monitor the sign of the level set function at each grid point, a time-explicit implementation is required. The standard narrow band approach is both time-explicit and computationally fast, so it represents an ideal framework for our algorithm. Second, we observe that the topology of the implicit contour is determined by the sign of the level set function, not its particular value. Therefore, the level set function is free to change its value in order to refine the position of the implicit contour at a *subpixel resolution*. In particular, despite the use of digital topology principles to control topology, the accuracy of the deformable model itself is still at the same subpixel level that is possible with standard geometric deformable models.

Explicit Contour Topology. We have now presented the basic notions describing how to relate the topology of the implicit contour to the discrete level set function and how to evolve the level set function in order to preserve topology. It is also important that we be able to *reconstruct* an explicit contour of the zero level set—a curve in 2D and a surface in 3D—and to guarantee that this reconstructed contour has the same topology as the digital object's boundary.

In a subsequent section, we describe how to modify a basic marching algorithm in order to produce an explicit contour having the same topology as the underlying digital object's boundary. It is this explicit model that we visualize (see Section 4), and that we use to characterize the topology of the evolving geometric deformable model. In particular, the topology of a given distinct contour can be summarized using its Euler characteristics χ, which given an explicit model can be computed using

$$\chi = N_V - N_E + N_F,$$

where N_V is the number of vertices, N_E is the number of edges, and N_F is the number of faces [48]. Note that N_F is always zero for 2D curves.

In principle, it is possible to *monitor* topological changes that are taking place during evolution of the level set function by counting the number of distinct contours and evaluating their Euler characteristics. The result of this computation cannot be used to *control* the topology since it is a global property of the contour(s), but it can be used to *verify* that a topology preserving mechanism is actually working properly. We used this computation in our experiments (see Section 4) to verify that both the evolving contour and the final contour had the correct topology. It is not necessary, in general, however, to compute the Euler characteristics in order to run TGDM.

3.2 Digital Topology

A 2D (respectively, 3D) digital (i.e., binary) image $V \subset \mathcal{Z}^2$ (respectively, $\mathcal{Z}^3$) is defined as a square (respectively, cubic)

4. Note that, by our convention, a sign change also happens if a zero value becomes positive or vice versa.

array of lattice points. The topology of a digital image depends on a pair of digital connectivities, one for the foreground and one for the background. We follow the conventional definition of *n-neighborhood* and *n-connectivity*, where $n \in \{4, 8\}$ in 2D and $n \in \{6, 18, 26\}$ in 3D [29]. 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 comprising 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, $(6, 18)$, $(18, 6)$, $(6, 26)$, and $(26, 6)$ are four pairs of compatible connectivities. The following definitions are from [31] and [47].

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

$$N_n^k(\mathbf{x}, X) = \cup\{N_n(\mathbf{y}) \cap N_M^*(\mathbf{x}) \cap X, \mathbf{y} \in N_n^{k-1}(\mathbf{x}, X)\},$$

where $M = 8$ in 2D and $M = 26$ in 3D.

Definition 2 (Topological Numbers). *Let $X \subset V$ and $\mathbf{x} \in V$. The topological numbers of the point $\mathbf{x}$ relative to the set X are: $T_4(\mathbf{x}, X) = \#\mathcal{C}_4(N_4^2(\mathbf{x}, X))$ and $T_8(\mathbf{x}, X) = \#\mathcal{C}_8(N_8^1(\mathbf{x}, X))$ in 2D; and*

$$T_6(\mathbf{x}, X) = \#\mathcal{C}_6(N_6^2(\mathbf{x}, X)), T_{6^+}(\mathbf{x}, X) = \#\mathcal{C}_6(N_6^3(\mathbf{x}, X)),$$

$$T_{18}(\mathbf{x}, X) = \#\mathcal{C}_{18}(N_{18}^2(\mathbf{x}, X)),$$

and $T_{26}(\mathbf{x}, X) = \#\mathcal{C}_{26}(N_{26}^1(\mathbf{x}, X))$ in 3D, where $\#$ denotes the cardinality of a set.

Intuitively, a n-connected neighbor of point $\mathbf{x}$ belongs to its geodesic neighborhood $N_n^k(\mathbf{x}, X)$ if there is a path in X of length no greater than k between the neighbor and the given point. The topological numbers are the numbers of connected components within certain geodesic neighborhoods. We note that, in the above definition of topological numbers in the 3D case, there are two notations for six-connectivity. This follows the convention introduced in [31], wherein the notation "6+" implies six-connectivity whose dual connectivity is 18, while the notation "6" implies six-connectivity whose dual connectivity is 26. This distinction is needed in order to correctly compute topological numbers under six-connectivity, and does not imply a different definition of connectivity.

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 a digital object does not change the object topology. It is proven in [31] that a point x is *simple* if and only if $T_n(\mathbf{x}, X) = 1$ and $T_{\bar{n}}(\mathbf{x}, \bar{X}) = 1$, where $(n, \bar{n})$ is a pair of compatible connectivities. In other words, characterization of a simple point requires only the computation of two topological numbers. These numbers can be computed using connected component labeling inside the $3 \times 3(\times 3)$ neighborhood of the candidate point.

3.3 Topology-Preserving Narrow Band Algorithm

In this section, we present the implementation of TLSM. The implementation consists of a subtle but important modification to the standard narrow band algorithm, which keeps the topology of the contour defined by the zero level set unchanged during the entire evolution. Two important questions that remain are considered in subsequent sections: 1) how does one create a topologically correct explicit representation of the final (or evolving) contour and 2) what are the convergence properties of the geometric deformable model implemented using TLSM?

In the following algorithm, it is convenient to store a binary-valued indicator function $B(\cdot)$, defined on the digital grid. For a grid point $\mathbf{x}_i$, $B(\mathbf{x}_i)$ equals 1 if $\Phi(\mathbf{x}_i, t_m) \leq 0$, and equals 0 otherwise, where t_m is the last time the point $\mathbf{x}_i$ is visited. The array $B(\cdot)$ is initialized by $\Phi(\cdot, 0)$, and is updated whenever the level set function Φ undergoes a sign change at a grid point $\mathbf{x}_i$. The sign change is computed using the following sign function definition, which reflects our convention that a zero valued grid point belongs to the interior of the zero level set:

$$\text{sign}(x) = \begin{cases} 1, & \text{if } x \leq 0; \\ -1, & \text{if } x > 0. \end{cases} \tag{10}$$

The algorithm is summarized below. Here, $\mathbf{x}_i$ is used to denote a general grid point and $\mathbf{y}_i$ denotes a narrow band point.

Algorithm 2 (Topology-Preserving Level Set Method)

1. *Initialize*—Set $m = 0$ and $t_m = 0$. Initialize $\Phi(\cdot, 0)$ to be the signed distance function of the initial contour. Initialize the binary indicator function B.
2. *Build the Narrow Band*—Find all grid points $\mathbf{y}_i, i \in \{1, \ldots, Q\}$ such that $|\Phi(\mathbf{y}_i, t_m)| < W_{\text{nb}}$, where W_{nb} is the user-specified narrow band width, and Q denotes the total number of narrow band points.
3. *Update*—For $i = 1, \cdots, Q$, compute the level set function at the narrow band point $\mathbf{y}_i$ at time $t_{m+1} = t_m + \Delta t$ by:
   (a) Using (9), compute $\Phi_{\text{temp}}(\mathbf{y}_i) = \Phi(\mathbf{y}_i, t_m) + \Delta t \Delta\ \Phi(\mathbf{y}_i, t_m)$.
   (b) If $\text{sign}(\Phi_{\text{temp}}(\mathbf{y}_i)) = \text{sign}(\Phi(\mathbf{y}_i, t_m))$, then set $\Phi(\mathbf{y}_i, t_{m+1}) = \Phi_{\text{temp}}(\mathbf{y}_i)$, keep $B(\mathbf{y}_i)$ unchanged, and go to Step 3(f). Otherwise continue to Step 3(c).
   (c) Compute the topological numbers $T_n(\mathbf{y}_i, X)$ and $T_{\bar{n}}(\mathbf{y}_i, \bar{X})$, where $(n, \bar{n})$ is the chosen digital connectivity pair, $X = \{\mathbf{x}_i | B(\mathbf{x}_i) = 1\}$, and $\bar{X} = \{\mathbf{x}_i | B(\mathbf{x}_i) = 0\}$.
   (d) If the point is simple—i.e., $T_n(\mathbf{y}_i, X) = T_{\bar{n}}(\mathbf{y}_i, \bar{X}) = 1$—then set $\Phi(\mathbf{y}_i, t_{m+1}) = \Phi_{\text{temp}}(\mathbf{y}_i)$, $B(\mathbf{y}_i) = (B(\mathbf{y}_i) + 1) \bmod 2$, and go to Step 3(f). Otherwise continue to Step 3(e).
   (e) Point $\mathbf{y}_i$ is not simple. To preserve the topology, we do not allow the sign change and set $\Phi(\mathbf{y}_i, t_{m+1}) = \epsilon \cdot \text{sign}(\Phi(\mathbf{y}_i, t_m))$, where ϵ is a small positive number. Note that $B(\mathbf{y}_i)$ remains unchanged.
   (f) Increase i. If $i > Q$, go to Step 4.
4. *Reinitialize*—If the zero level set of $\Phi(\cdot, t_{m+1})$ is near the boundary of the current narrow band, reinitialize $\Phi(\cdot, t_{m+1})$ to be the signed distance function of its zero level set.
5. *Convergence Test*—Test whether the zero level set has stopped moving. If yes, stop; otherwise, set $m = m + 1$. If reinitialization was performed in Step 4, then go back to Step 2 to rebuild the narrow band; otherwise, go back to Step 3.

Compared with Algorithm 1, the TLSM algorithm differs only in the Update step, which performs a simple point criterion check whenever the level set function is going to change sign at a grid point. The sign change is prohibited if the point is not a simple point, and the evolution of the level set function at that point is limited. One might ask how this limiting operation would affect the convergence property of the new model. We will show later that the above algorithm is a direct analog of the gradient-descent-with-bending algorithm in the literature of constrained optimization [49] and, thus, is guaranteed to converge to a constrained optimum.

We would like to point out that there can be some arbitrariness in the specific result of the algorithm depending on the order in which the points are visited in the narrow band. This situation is also present in skeletonization algorithms where the result depends on the order of simple point removal [50]. The problem is not as significant here, however, as in skeletonization, since the overall motion of the deforming contour is controlled by the internal and external forces. The simple point criterion only takes effect at locations where topological changes are otherwise going to occur, and these locations ordinarily comprise a very small portion of the overall contour. Still, we have compared the results of two different orderings for visiting the narrow band points. In one case, we ordered the points by the magnitude of their external force, and in the other case, by a natural ordering that "rasters" through the coordinates of the points. The difference was trivial and did not favor either approach. In the experiments reported herein, we visit the narrow band points by the natural "raster" ordering of their coordinates.

3.4 Connectivity Consistent Isocontour Algorithms

The design of a level set method is not complete without studying the isocontour algorithm that produces an explicit representation of the final contour from the embedding level set function. The choice of a suitable isocontour algorithm is especially critical for the new topology-preserving models where the algorithm must faithfully recover the topology of the implicit contour from the discrete samples of the level set function. In the following discussion, we will focus on the 3D case where we modify the standard *marching cubes* (MC) algorithm and arrive at a new *connectivity consistent marching cubes* (CCMC) algorithm that is consistent with our topology preservation principle. The 2D case is a simplified version and is referred to as the *connectivity consistent marching squares* (CCMS) algorithm.

The MC algorithm is a standard isosurface algorithm that produces a triangulated surface whose vertices lie on the edges of the cubic lattice [27]. As shown in Fig. 2, 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 each ambiguous case. A well-accepted criterion is that the surface tiling should correctly reflect the topology of the true underlying implicit surface. Under the assumption that the embedding function is densely sampled and approximately linear on each cube, *face saddle points* and *body saddle points* can be used to produce isosurfaces that are topologically equivalent to the embedded implicit surfaces [51]. 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 vanish.

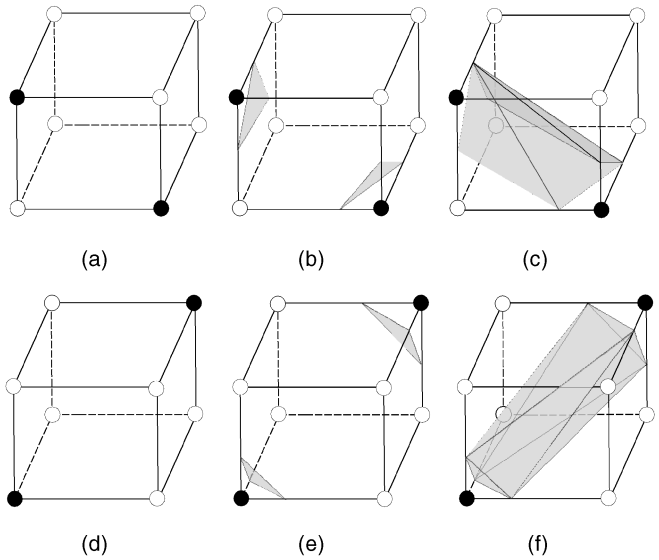


Fig. 2. (a) An ambiguous face; (b) and (c) are two possible tilings. (d) An ambiguous cube; (e) and (f) are two possible tilings.

From the discussion in Section 3.1, it is clear that what we need in this paper is an isosurface algorithm that can correctly recover the *digital* topology embedded in the level set function, which depends on the predefined digital connectivity rule. 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 (which gives subpixel resolution), but which surface tiling to choose depends on the given digital connectivity. In particular, we choose the tilings in Figs. 2c and 2e for the corresponding ambiguous cases respectively if the black points are assumed to be 18-connected while the white points are six-connected. If the black points are assumed to be 26-connected, then Figs. 2c and 2f should be used instead. As can be expected, the tilings for unambiguous cases are the same as in the standard MC algorithm.

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. 2a). The correct tiling should separate the white points while connect the black ones if the black points are eight-connected, and vice versa.

After the level set iterations have converged, we extract the final contour using the CCMS (2D) or CCMC (3D) algorithm. As stated above, the 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 contour.[5] 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 $-\epsilon$.

5. This is one of several major artifacts that exist in most existing isocontour software.

3.5 Convergence Analysis

To analyze the convergence property of TLSM-based geometric deformable models (i.e., a TGDM), we focus on the case where the model is derived from an energy minimization framework, for example, the geodesic deformable model. In the original formulation of the geodesic deformable model, the energy to be minimized is defined as a functional on the family of explicitly parameterized contours. The level set evolution equation is then derived by applying the level set method. By adopting the techniques presented in [52], [40], we can derive the evolution equation of the geodesic deformable model directly from an energy functional defined on the level set function itself. We can then show that the corresponding TGDM algorithm in this case is a constrained gradient descent algorithm, and is guaranteed to converge to a constrained optimal point of the energy functional. Assume that the level set function $\Phi(\mathbf{x}), \mathbf{x} \in \Omega$ where $\Omega \subset \mathcal{R}^2$ (respectively, $\mathcal{R}^3$) is Lipschitz-continuous. Then, it can be proved by the co-area formula [53] that the length (respectively, area) of the zero level set of Φ is given by:

$$L(\Phi) = \int_{\Omega} \delta(\Phi(\mathbf{x}))|\nabla\Phi(\mathbf{x})|d\mathbf{x},$$

where $\delta(\cdot)$ is the one-dimensional Dirac delta function. Similarly, the weighted length or area L_g under an image derived metric $g(\mathbf{x})$ is given by

$$L_g(\Phi) = \int_{\Omega} \delta(\Phi(\mathbf{x}))|\nabla\Phi(\mathbf{x})|gd\mathbf{x}.$$

To make the equations shorter, in the following derivation we omit the function argument $\mathbf{x}$ when there is no potential for confusion.

The Frechet derivative of L_g with respect to $\Phi(\mathbf{x})$ in the direction $h(\mathbf{x})$, which is denoted by $dL_g(\Phi, h)$, can be computed as

$$dL_g(\Phi, h) = \int_{\Omega} h\delta'(\Phi)|\nabla\Phi|gd\mathbf{x} + \int_{\Omega} \delta(\Phi)g\frac{\nabla\Phi \cdot \nabla h}{|\nabla\Phi|}d\mathbf{x},$$

where $\delta'(\cdot)$ denotes the first derivative of the delta function.

Applying Green's formula [54] to the second term yields

$$dL_g(\Phi, h) = \int_{\Omega} h\delta'(\Phi)|\nabla\Phi|gd\mathbf{x} + \oint_{\partial\Omega} h\delta(\Phi)g\frac{\nabla\Phi \cdot \vec{n}}{|\nabla\Phi|}ds - \int_{\Omega} h\nabla \cdot \left(\delta(\Phi)g\frac{\nabla\Phi}{|\nabla\Phi|}\right)d\mathbf{x},$$

where $\nabla\cdot$ is the divergence operator, $\vec{n}$ is the normal vector to the boundary, and ds is a differential element on the boundary.

Since $\nabla\Phi \cdot \vec{n} = \partial\Phi/\partial\vec{n}$ and

$$\nabla \cdot \left(\delta(\Phi)g\frac{\nabla\Phi}{|\nabla\Phi|}\right) = g\delta'(\Phi)|\nabla\Phi| + \delta(\Phi)\nabla \cdot \left(g\frac{\nabla\Phi}{|\nabla\Phi|}\right),$$

under the natural boundary condition $\partial\Phi/\partial\vec{n} = 0$ we get

$$dL_g(\Phi, h) = -\int_{\Omega} \delta(\Phi)\nabla \cdot \left(g\frac{\nabla\Phi}{|\nabla\Phi|}\right)hd\mathbf{x} = < -\delta(\Phi)\nabla \cdot \left(g\frac{\nabla\Phi}{|\nabla\Phi|}\right), h >,$$

where $< \cdot, \cdot >$ denotes inner product in the L^2 sense. From the Schwartz inequality [54], it is clear that the direction that reduces the energy functional L_g most rapidly, that is, the steepest descent direction h_s, is given by

$$h_s = \delta(\Phi)\nabla \cdot \left(g\frac{\nabla\Phi}{|\nabla\Phi|}\right) = \delta(\Phi)\left\{\frac{\nabla g \cdot \nabla\Phi}{|\nabla\Phi|} + g\nabla \cdot \left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\right\},$$

where the second equality follows from a vector calculus identity. Thus, starting from an initial estimate $\Phi(\mathbf{x}, 0)$, the gradient descent algorithm with an infinitesimal time step δt gives the level set evolution equation as

$$\frac{\partial\Phi(\mathbf{x}, t)}{\partial t} = h_s(\mathbf{x}) = \delta(\Phi(\mathbf{x}, t))\left\{\frac{\nabla g(\mathbf{x}) \cdot \nabla\Phi(\mathbf{x}, t)}{|\nabla\Phi(\mathbf{x}, t)|} + g(\mathbf{x})\nabla \cdot \left(\frac{\nabla\Phi(\mathbf{x}, t)}{|\nabla\Phi(\mathbf{x}, t)|}\right)\right\}, \tag{11}$$

such that the family of level set functions $\Phi(\cdot, t)$ will converge to the (local) minimum of the energy functional L_g as t goes to infinity.

We can see that the only difference between (6) and (11) is that the scale factor $\delta(\Phi(\mathbf{x}, t))$ in (11) is replaced by $|\nabla\Phi(\mathbf{x}, t)|$ in (6), which corresponds to extending the evolution equation to all the level sets of Φ [52]. Since the energy functional L_g depends only on the zero level set of Φ, we note that (6) also gives a steepest descent minimization of L_g.

When topology preservation is required, the gradient descent process must be constrained to the admissible set (or feasible domain) of level set functions that satisfy the topology constraint. In our case, this feasible domain comprises level set functions whose zero level sets share the prescribed model topology. From an optimization viewpoint, we can think of a single step of the gradient descent algorithm as a modification of the entire level set function in order to produce a new level set function. If that new function were outside of the feasible domain (i.e., its zero level set did not have the correct topology), then one possible modification to the algorithm would be to reduce the step size until the modified function remained in the feasible domain. Unfortunately, this simple strategy has been shown in the literature on constrained optimization to suffer from the "jamming" effect and nonconvergence [49], [55]. To avoid the jamming effect (and thereby to guarantee convergence), McCormick [49] proposed the *constrained gradient descent with bending* algorithm. The key idea is that instead of reducing the whole step size, which is equivalent to multiplying the descent vector by a small constant, only that component of the descent vector which leads outside the feasible domain should be reduced (or truncated) while the other components should keep the usual step size. This strategy has been shown to avoid the jamming effect and to always converge to a constrained stationary point, i.e., a Kuhn-Tucker point [55].

Our TGDM algorithm is an adaptation of McCormick's approach. To see this, we note that an individual component of the gradient descent vector in the geodesic deformable model is exactly the force function evaluated at an individual grid point. Those components that lead to a

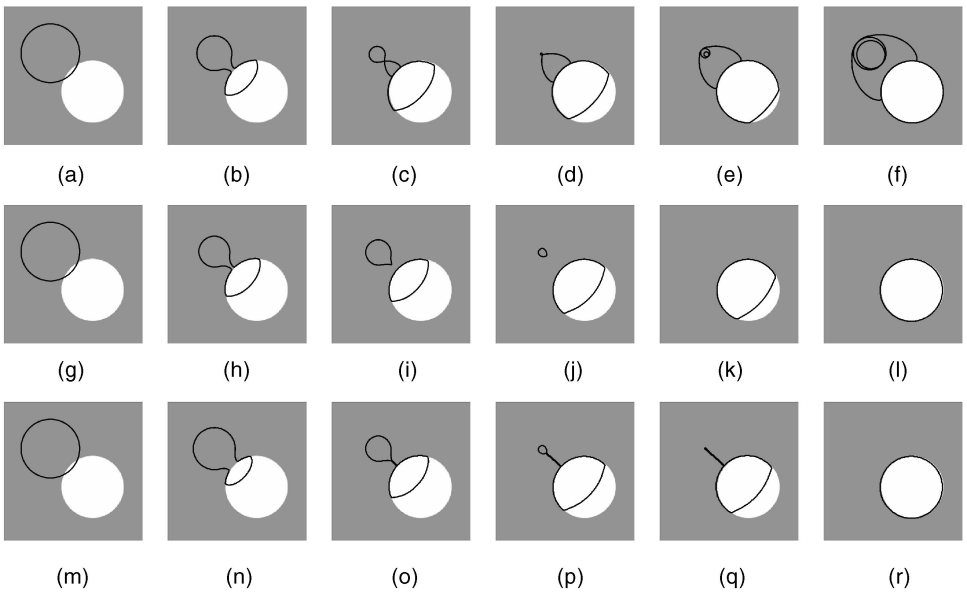


Fig. 3. A 2D phantom illustrating the self-intersection problem of PDM. (a)-(f) Propagation of the PDM contour at several time steps. (g)-(l) Evolution of the SGDM contour. (m)-(r) Evolution of the TGDM model.

violation of the topology constraint and, therefore, would move the whole level set function out of the feasible domain, can be determined by the simple point criterion. Step 3(e) in the topology-preserving narrow band algorithm thus corresponds to the truncating of the component of the gradient descending vector that would lead to movement out of the feasible domain; the other components remain unchanged. This is exactly the *bending* of the gradient descent direction as described in [49].

As pointed out previously, there can be some arbitrariness in the specific result of the algorithm depending on the order in which the grid points are visited in the narrow band. This arbitrariness reflects the fact that the feasible domain of the topology-constrained minimization problem is nonconvex. As a result, at a concave corner of the feasible domain, there can be more than one possible direction to bend the original gradient descent vector. Which direction the bending actually occurs then depends on the order in which the grid points are visited.

We note that as the unconstrained model can only be guaranteed to converge to a local optimum depending on the initialization, the TGDM may also only converge to a constrained local optimum. We also note that in a general geometric deformable model where the evolution equation does not come from an energy minimization formulation, the above optimality and convergence analysis does not apply. But, from our experience and as demonstrated in the presented experiment results, the TGDM algorithm shares the same convergence property as its nonconstrained counterpart.

4 Results

In this section, we present several experiments which apply the new topology-preserving geometric deformable models in 2D and 3D. Since the new models can be obtained from existing geometric deformable models by applying the TLSM narrow band implementation, we will refer to the original models without topology constraint (implemented by the standard narrow band algorithm) as *standard geometric deformable models* (SGDM's) and the corresponding (i.e., with the same set of force terms) topology-preserving models as *topology-preserving geometric deformable models* (TGDM's). When a parametric deformable model with a similar set of force terms is also compared, it will be referred to as the *parametric deformable model* (PDM). Note that for the TGDM, the CCMS or CCMC algorithm must be used in order to correctly extract the final curves or surfaces from the level set function. The SGDM, on the other hand, requires a standard isocontour algorithm, preferably one that uses face saddle points in 2D and both face and body saddle points in 3D [51]. 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.

2D Experiments. Fig. 3 shows a 2D example that compares the behavior of PDM, SGDM, and TGDM. All three models apply a curvature force as the smoothing internal force and a region force that expands inside the white circular cell and contracts outside. The top row of Fig. 3 shows the propagation of the PDM contour at several time steps starting from the initialization shown in Fig. 3a. The curve intersects with itself and then goes unstable because the normal direction gets flipped over after the curve self-intersects and the region force begins to push the curve in the wrong direction. The SGDM curve (the middle row) changes topology twice, first splitting in Fig. 3i and then losing one curve in Fig. 3k. On the other hand, the TGDM curve maintains the same topology throughout its evolution and does not suffer from the self-intersection problem. It is apparent that in this case the SGDM and the TGDM produced the same final contour.

The second 2D experiment, shown in Fig. 4, used the same phantom but a different initialization. A variation of the geodesic deformable contour model of (6) was used as

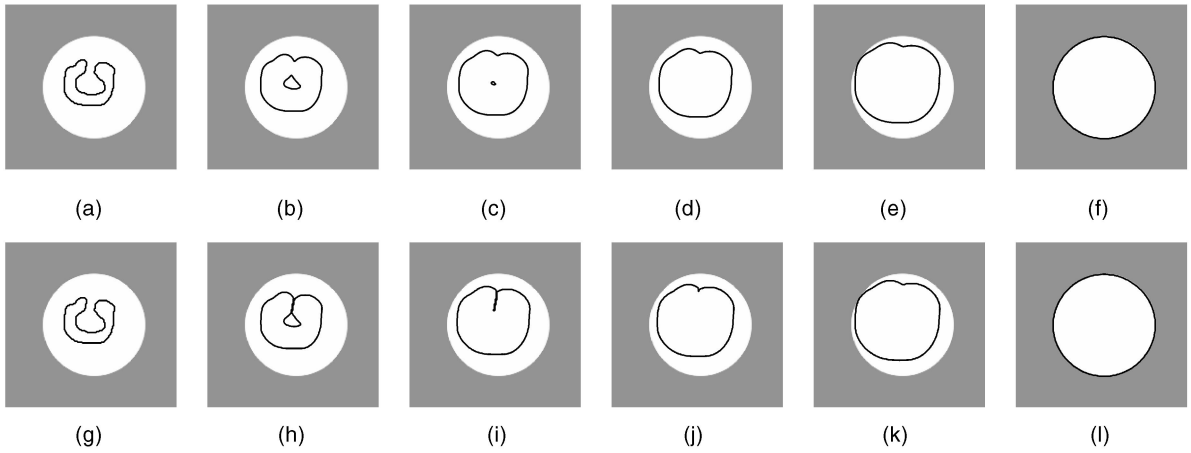


Fig. 4. Illustration of TGDM reaching the same global optimum as SGDM. (a)-(f) SGDM result and (g)-(l) TGDM result.

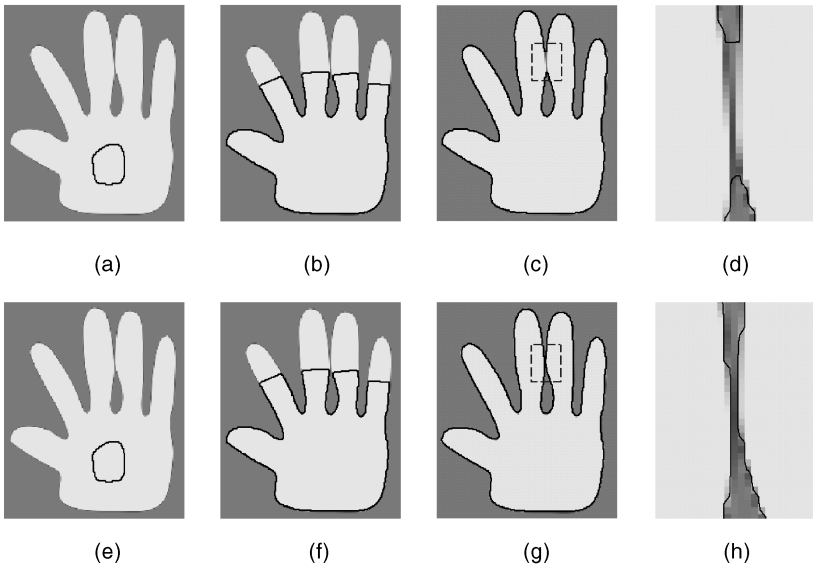


Fig. 5. Segmentation of a hand phantom using both SGDM (top row) and TGDM (bottom row).

the SGDM model, where an additional expansion force $cg(\mathbf{x})|\nabla\Phi(\mathbf{x},t)|$ (with c constant) was added to increase the speed of convergence [8]. The final contour thus corresponds to the solution of the geodesic energy minimization problem. The corresponding TGDM model was derived from the SGDM by imposing the topology-preserving constraint. Comparing the two rows of Fig. 4, it can be seen that the two models achieve the same global optimum through different optimization paths: the SGDM curve changed topology twice, whereas the TGDM curve maintains the same topology throughout the entire evolution. It is important to notice that the TGDM curve is able to evolve out of an unfavorable configuration formed during the early stages, and that the topology constraint takes effect early, but is released automatically later in the evolution. This demonstrates that the TGDM curve was not "jammed" by the topology constraint into the configuration of Fig. 4h or Fig. 4i; instead, it successfully converges to the global optimum.

Fig. 5 shows another 2D example in which the SGDM and the TGDM geodesic active contour models used in previous experiment were applied again to find the boundary of a hand-shaped object. The original image (220×190 pixels) and the initial curve are shown in both Fig. 5a and Fig. 5e. Figs. 5b and 5c show the SGDM contour at an intermediate and the final stage, respectively. Because the two middle fingers touch, the initial curve changes topology and splits into two separate curves as the final result (a larger outer curve and a disjoint inner curve as shown in Fig. 5c and zoomed up in Fig. 5d). We note that the two middle fingers in the hand become one "finger" with a hole in it in the final segmentation, which is obviously an undesirable result. The corresponding deformations of the TGDM contour are illustrated in Figs. 5f and 5g. 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. 5h.

As a final 2D example, we apply an SGDM model and the corresponding TGDM model to find the boundary of two adjacent bone cells in a CT image. The deformable model we adopt here is the binary-flow model proposed in [14]. The model applies a dynamic region force which tries to maximally separate the mean of the region encircled by the evolving contours from that of its complement. The

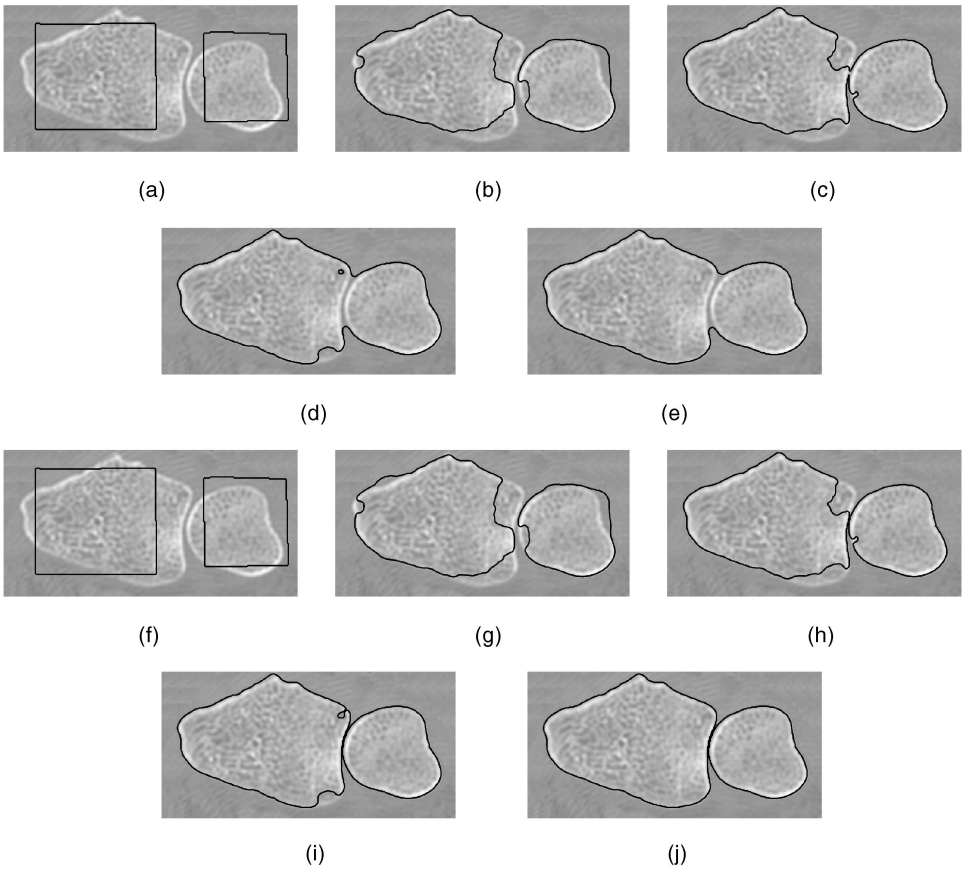


Fig. 6. Segmentation of a carpal bones CT image using both SGDM and TGDM. (a) and (f) The initialization. (b)-(e) Evolution of the SGDM contour. (g)-(j) Evolution of the TGDM contour. (Original image courtesy of B. Kimia.)

curvature force is also used as a regularization force to counteract with the effect of image noise. Figs. 6a and 6f show the image overlaid with the initial curves. Without the topology constraint, the two separate curves merge at the weak gap between the two bone cells and one single contour that encloses both bones is produced as the final result. Again, the TGDM curves keep separated throughout the evolution and correctly find the boundary of each cell.

3D Experiments. In 3D, a promising area of application of the new topology-preserving deformable model is the human brain mapping. Some preliminary results have been reported in a conference paper [35]. More detailed work will appear in a separate paper. In the following, we present the results from two 3D experiments for illustration purposes.

As the first 3D example, we applied a 3D version of the geometric deformable model of (8) to find the boundary surface of the 3D object depicted in Fig. 7a. The object is actually a piece of a white matter segmented from a magnetic resonance (MR) brain image. Due to data noise, the white matter piece has a handle, 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. 7b.

Figs. 7c and 7d are the final surfaces obtained by SGDM. The two results are the same since standard geometric deformable models are insensitive to initialization. The final surface has a handle, however, which is the incorrect topology. With the sphere as the initialization, TGDM gives the final surface shown in Fig. 7e, and with the ellipsoid, it gives the result shown in Fig. 7f. Both surfaces have the

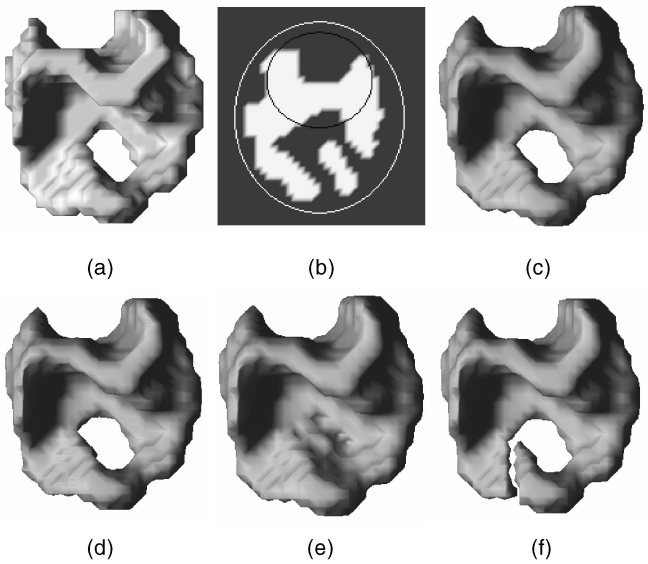
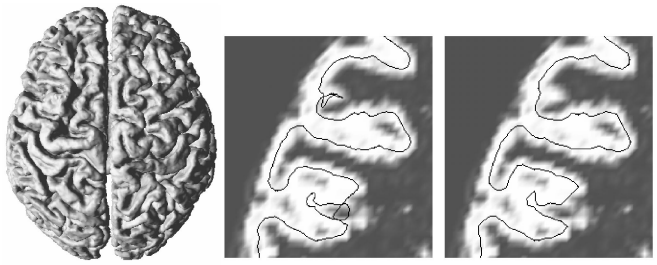
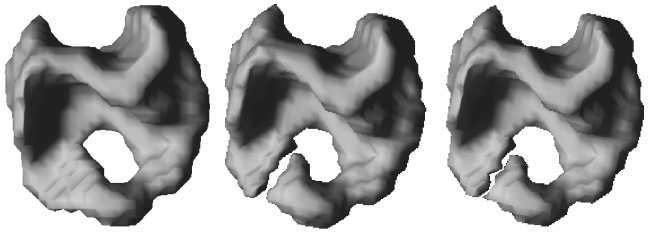


Fig. 7. (a) A 3D phantom, (b) large sphere and small ellipsoid initializations, (c) SGDM result from sphere, (d) SGDM result from ellipsoid, (e) TGDM result from sphere, and (f) TGDM result from ellipsoid.



(a) (b) (c)

Fig. 8. (a) Result of cortical surface reconstruction. (b) Self-intersection from PDM and (c) no intersection with TGDM.



(a) (b) (c)

Fig. 9. (a) The final result without topology constraint (SGDM), (b) initialization by topology correction, and (c) final result with TGDM.

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*—a handle in the background image—through the original object, while the ellipsoid initialization makes a cut in the handle. The dependency of TGDM on initialization is discussed in detail in the next section.

Our second 3D experiment applies SGDM, TGDM, and a parametric deformable surface model (PDM) to extract the central cortical surface from an initial fuzzy segmentation of a brain MR image volume. We used exactly the same initialization (a topologically correct surface near the gray-matter/white-matter interface), the same external forces, and similar internal forces for the geometric deformable models and the PDM. The results are presented in Fig. 8. Fig. 8a shows the final surface extracted from the parametric model. The SGDM and TGDM surfaces look identical at this level of detail, but on close examination, there are important differences. The parametric model result, for example, has self-intersections as shown in Fig. 8b, while the TGDM surface does not, as shown in Fig. 8c. Also, the SGDM result has 40 handles, whereas both the parametric model result and the TGDM result have no handles and, hence, are topologically equivalent to spheres. 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

Several issues are discussed in this section, including the dependency of the final segmentation results on different initializations, the computational complexity of imposing the topology constraint, and some related investigations in the literature.

The example shown in Fig. 7 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 issue of topology preservation. At present, we have no formulation of an optimality criterion that would choose one of these solutions over the other. This situation is not atypical in deformable models, where the particular initialization very often determines the exact details of the final solution. As a step towards reducing the dependency on particular initialization, one can drop the topology constraint initially, that is, apply the standard geometric model to achieve an initial unconstrained optimal solution. After the unconstrained optimum is obtained, one can then apply a topology correction method to "project" the temporary solution back to the feasible domain, and start the constrained deformation from this "better" initialization. As one example, we applied the topology correction method of [25] on the SGDM result of Fig. 7c (which is also shown in Fig. 9a for clarity) to get the initialization shown in Fig. 9b. The TGDM model then produced the final result as shown in Fig. 9c. The new result is similar to that of Fig. 7f, but it is now the unique solution, independent of initialization. We note that such a topology correction (projection) method is not generally available for all topologies (the particular method can only be used to achieve a spherical topology), and when such an approach is taken, the initialization is determined by the optimality criterion applied in the topology correction method.

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 narrow band implementation. For the phantom experiments, the time difference between standard and new geometric models are 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.

A related work is the shock detection method of [57], [58]. It is known from the Morse theory [59] that the implicit contour (zero level set of a level set function) undergoes topology changes if a critical point (extremal or saddle point) of the level set function, known as a *shock* point in [57], [58], passes through the zero level set, or in other words, it changes sign (see also [60]). Although the shock detection algorithm is promising in analyzing static shapes, it is time-consuming and unreliable in detecting and tracing sign changes of all shocks of a level set function evolving under a general velocity field, especially in 3D. One reason is that the level set function is sampled on discrete grids while its critical points are usually located between grid lines. In the topology-preserving mechanism proposed in this paper, the topology change is directly correlated with sign-change of the level set function itself on the grid points, which provides a simple way to detect and prevent topology changes. However, the level set function itself is still evolving continuously, thus subpixel accuracy is maintained.

Another related work is the skeletally-coupled deformable model proposed by Sebastian et al. [56] for 2D carpal

bone image segmentation. In this work, each bone cell is represented by a distinctly labeled region, and no two regions are allowed to merge during a seeded region growing. This approach only deals with one type of topological change—the merging of two disjoint regions. It cannot be easily generalized to deal with the splitting of a single region like the example shown in Fig. 5, nor can it be generalized to deal with one single region developing a handle in 3D. Overall, we believe that the most efficient and straightforward way to guarantee topology preservation in a deformable model algorithm is by applying the simple point criterion, as we have proposed herein.

6 Conclusion

In summary, we have developed a novel topology-preserving level set method from which we derived a new class of 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 the standard narrow band implementation of the traditional level set method. We have also shown that, for energy minimizing geometric deformable models, their topology-preserving counterpart is a special constrained gradient descent algorithm that does not suffer from the jamming effect and is guaranteed to converge to a constrained optimal point of the energy functional. Several 2D and 3D experiments were conducted to show the success of the new models and illustrate their potential applications.

Acknowledgments

This work was supported in part by the US National Science Foundation/ERC grant CISST#9731748 and NIH/NINDS grant R01NS37747.

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models," *Int'l J. Computer Vision,* vol. 1, pp. 312-333, 1988.

[2] L.D. Cohen, "On Active Contour Models and Balloons," *CVGIP: Image Understanding,* vol. 53, pp. 211-218, 1991.

[3] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham, "Active Shape Models—Their Training and Application," *CVGIP: Image Understanding,* vol. 61, no. 1, pp. 38-59, 1995.

[4] C. Xu and J.L. Prince, "Snakes, Shapes, and Gradient Vector Flow," *IEEE Trans. Image Processing,* vol. 7, no. 3, pp. 359-369, 1998.

[5] C. Chesnaud, P. Réfrégier, and V. Boulet, "Statistical Region Snake-Based Segmentation Adapted to Different Physical Noise Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, pp. 1145-1157, 1999.

[6] V. Caselles, F. Catte, T. Coll, and F. Dibos, "A Geometric Model for Active Contours in Image Processing," *Numerische Mathematik,* vol. 66, pp. 1-31, 1993.

[7] R. Malladi, J.A. Sethian, and B.C. Vemuri, "Shape Modeling with Front Propagation: A Level Set Approach," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, pp. 158-175, 1995.

[8] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," *Int'l J. Computer Vision,* vol. 22, pp. 61-79, 1997.

[9] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert, "Minimal Surfaces Based Object Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, pp. 394-398, 1997.

[10] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Gradient Flows and Geometric Active Contours," *Proc. Int'l Conf. Computer Vision '95,* pp. 810-815, 1995.

[11] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, "Conformal Curvature Flows: From Phase Transitions to Active Vision," *Archive for Rational Mechanics and Analysis,* vol. 134, pp. 275-301, 1996.

[12] A. Yezzi, S. Kichenassamy, P. Olver, and A. Tannenbaum, "A Geometric Snake Models for Segmentation of Medical Imagery," *IEEE Trans. Medical Imaging,* vol. 16, pp. 199-209, 1997.

[13] K. Siddiqi, Y.B. Lauziere, A. Tannenbaum, and S.W. Zucker, "Area and Length Minimizing Flow for Shape Segmentation," *IEEE Trans. Image Processing,* vol. 7, pp. 433-443, 1998.

[14] A. Yezzi, A. Tsai, and A. Willsky, "A Statistical Approach to Snakes for Bimodal and Trimodal Imagery," *Proc. Int'l Conf. Computer Vision '99,* pp. 898-903, 1999.

[15] N. Paragios and R. Deriche, "Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, pp. 1-15, 2000.

[16] M. Leventon, E. Grimson, and O. Faugeras, "Statistical Shape Influence in Geodesic Active Contours," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 316-322, June 2000.

[17] T. McInerney and D. Terzopoulos, "Deformable Models in Medical Image Analysis: A Survey," *Medical Image Analysis,* vol. 1, no. 2, pp. 91-108, 1996.

[18] J.A. Sethian, *Level Set Methods and Fast Marching Methods.* Cambridge, UK: Cambridge Univ. Press, second ed., 1999.

[19] T. McInerney and D. Terzopoulos, "T-Snakes: Topology Adaptive Snakes," *Medical Image Analysis,* vol. 4, pp. 73-91, 2000.

[20] H. Delingette and J. Montagnat, "New Algorithms for Controlling Active Contour Shape and Topology," *Proc. European Conf. Computer Vision (ECCV 2000),* pp. 381-395, 2000.

[21] 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,* vol. 12, pp. 340-356, 2000.

[22] 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. Medical Imaging,* vol. 18, no. 6, pp. 467-480, 1999.

[23] D.W. Shattuck and R.M. Leahy, "Topologically Constrained Cortical Surfaces from MRI," *Proc. SPIE Conf.,* vol. 3979, pp. 747-758, Feb. 2000.

[24] 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. Medical Imaging,* vol. 20, no. 1, pp. 70-80, 2001.

[25] X. Han, C. Xu, U. Braga-Neto, and J.L. Prince, "Graph-Based Topology Correction for Brain Cortex Segmentation," *Proc. 17th Int'l Conf. Information Processing in Medical Imaging,* pp. 395-401, June 2001.

[26] S. Osher and J.A. Sethian, "Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations," *J. Computational Physics,* vol. 79, pp. 12-49, 1988.

[27] W.E. Lorensen and H.E. Cline, "Marching Cubes: A High-Resolution 3D Surface Construction Algorithm," *ACM Computer Graphics,* vol. 21, no. 4, pp. 163-170, 1987,

[28] G. Hermosillo, O. Faugeras, and J. Gomes, "Unfolding the Cerebral Cortex Using Level Set Methods," *Proc. Second Int'l Conf. Scale-Space Theories in Computer Vision,* pp. 58-69, 1999.

[29] T.Y. Kong and A. Rosenfeld, "Digital Topology: Introduction and Survey," *CVGIP: Image Understanding,* vol. 48, pp. 357-393, 1989.

[30] P.K. Saha and B.B. Chaudhuri, "Detection of 3D Simple Points for Topology Preserving Transformations with Application to Thinning," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, pp. 1028-1032, 1994.

[31] G. Bertrand, "Simple Points, Topological Numbers and Geodesic Neighborhoods in Cubic Grids," *Pattern Recognition Letters,* vol. 15, pp. 1003-1011, 1994.

[32] 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. Imaging Vision,* vol. 5, pp. 297-318, 1995.

[33] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny, "Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data," *Proc. SIGGRAPH '91* pp. 217-226, 1991.

[34] X. Han, C. Xu, and J.L. Prince, "A Topology Preserving Deformable Model Using Level Set," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 765-770, Dec. 2001.

[35] X. Han, C. Xu, D. Tosun, and J.L. Prince, "Cortical Surface Reconstruction Using a Topology Preserving Geometric Deformable Model," *Proc. Fifth IEEE Workshop Math. Methods in Biomedical Image Analysis (MMBIA '2001),* pp. 213-220, Dec. 2001.

[36] X. Han, C. Xu, and J.L. Prince, "A Topology Preserving Geometric Deformable Model and Its Application in Brain Cortical Surface Reconstruction," *Geometric Level Set Methods in Imaging, Vision, and Graphics,* S. Osher and N. Paragios, eds. Springer Verlag, 2003.

[37] J.A. Sethian, "A Fast Marching Level Set Method for Monotonically Advancing Fronts," *Proc. Nat'l Academy of Sciences,* vol. 93, pp. 1591-1595, 1996.

[38] J.N. Tsitsiklis, "Efficient Algorithm for Globally Optimal Trajectories," *IEEE Trans. Automatic Control,* vol. 40, no. 9, pp. 1528-1538, 1995.

[39] H. Tek and B.B. Kimia, "Image Segmentation by Reaction-Diffusion Bubbles," *Proc. Int'l Conf. Computer Vision,* pp. 156-162, 1995.

[40] T.F. Chan and L.A. Vese, "Active Contours without Edges," *IEEE Trans. Image Processing,* vol. 10, no. 2, pp. 266-277, 2001.

[41] C. Xu, A. Yezzi, and J.L. Prince, "A Summary of Geometric Level-Set Analogues for a General Class of Parametric Active Contour and Surface Models," *Proc. First IEEE Workshop Variational and Level Set Methods,* pp. 104-111, 2001.

[42] D. Adalsteinsson and J.A. Sethian, "The Fast Construction of Extension Velocities in Level Set Methods," *J. Computational Physics,* vol. 148, pp. 2-22, 1999.

[43] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky, "Fast Geodesic Active Contours," *IEEE Trans. Image Processing,* vol. 10, no. 10, pp. 1467-1475, 2001.

[44] J. Weickert, B.M.t.H. Romeny, and M.A. Viergever, "Efficient and Reliable Scheme for Nonlinear Diffusion Filtering," *IEEE Trans. Image Processing,* vol. 7, pp. 398-410, 1998.

[45] D.L. Chopp, "Computing Minimal Surfaces via Level Set Curvature Flow," *J. Computational Physics,* vol. 106, no. 1, pp. 77-91, 1993.

[46] D. Adalsteinsson and J.A. Sethian, "A Fast Level Set Method for Propagating Interfaces," *J. Computational Physics,* vol. 118, pp. 269-277, 1995.

[47] G. Bertrand, J.C. Everat, and M. Couprie, "Image Segmentation through Operators Based on Topology," *J. Electronic Imaging,* vol. 6, pp. 395-405, 1997.

[48] M.J. Greenberg and J.R. Harper, *Algebraic Topology: A First Course.* Mass.: Addison Wesley, 1981.

[49] G.P. McCormick, "Anti-Zig-Zagging by Bending," *Management Science,* vol. 15, pp. 315-320, 1969.

[50] C. Pudney, "Distance-Ordered Homotopic Thinning: A Skeletonization Algorithm for 3D Digital Images," *Computer Vision and Image Understanding,* vol. 72, pp. 404-413, 1998.

[51] B.K. Natarajan, "On Generating Topologically Consistent Isosurfaces from Uniform Samples," *Visual Computer,* vol. 11, no. 1, pp. 52-62, 1994.

[52] H.K. Zhao, T. Chan, B. Merriman, and S. Osher, "A Variational Level Set Approach to Multiphase Motion," *J. Computational Physics,* vol. 127, pp. 179-195, 1996.

[53] L.C. Evans and R.F. Gariepy, *Measure Theory and Fine Properties of Functions.* 1992.

[54] G. Strang, *Introduction to Applied Math.* Wellesley Cambridge Press, 1986.

[55] M.S. Bazaraa and C.M. Shetty, *Nonlinear Programming: Theory and Algorithms.* New York: John Wiley & Sons, 1979.

[56] T.B. Sebastian, H. Tek, S.W. Wolfe, J.J. Crisco, and B.B. Kimia, "Segmentation of Carpal Bones from 3D CT Images Using Skeletally Coupled Deformable Models," *Proc. Medical Image Computing and Computer Assisted Intervention Conf. '98,* pp. 1184-1194, Oct. 1998.

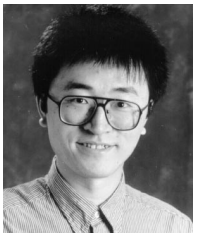
[57] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S. Zucker, "Shock Graphs and Shape Matching," *Int'l J. Computer Vision,* vol. 35, no. 1, pp. 13-32, 1999.

[58] K. Siddiqi, B.B. Kimia, A.R. Tannenbaum, and S. Zucker, "Shapes, Shocks and Wiggles," *Image Vision Computing,* vol. 17, pp. 365-373, 1999.

[59] J. Milnor, "Morse Theory," *Annals Math. Studies,* vol. 51, Princeton Univ. Press, 1963.

[60] B.T. Stander and J.C. Hart, "Guaranteeing the Topology of an Implicit Surface Polygonization for Interactive Modeling," *Proc. SIGGRAPH '97,* pp. 279-286, 1997.

Xiao Han received the BS degree in electrical engineering from the University of Science and Technology of China in 1994, the MEng degree from the National University of Singapore in 1997, and the MSE degree from the Johns Hopkins University in 2001. He is currently working toward the PhD degree in the Department of Electrical and Computer Engineering at the Johns Hopkins University. His research interests are in medical image processing, deformable models, and digital topology. He is a student member of the IEEE.



Chenyang Xu received the BS degree in computer science and engineering from the University of Science and Technology of China in 1993 and the MSE and PhD degrees in 1995 and 1999, respectively, from the Johns Hopkins University, all in electrical and computer engineering. He was a postdoctoral fellow and then an associate research scientist in both the Center for Imaging Science and the US National Science Foundation Engineering Research Center for Computer Integrated Surgical Systems and Technology at the Johns Hopkins University during 1999-2000. Since 2000, he has been a member of technical staff in the imaging and visualization department at Siemens Corporate Research, Inc. His research interests are in the areas of image processing, computer vision, medical imaging, deformable models, and brain mapping. He is a member of the IEEE.

Jerry L. Prince received the BS degree from the University of Connecticut in 1979 and the SM, EE, and PhD degrees in 1982, 1986, and 1988, respectively, from the Massachusetts Institute of Technology, all in electrical engineering and computer science. He has worked as an engineer at the Brigham and Women's Hospital, MIT Lincoln Laboratories, and The Analytic Sciences Corporation (TASC). He joined the faculty at the Johns Hopkins University in 1989, where he is currently the William B. Kouwenhoven Professor in the Department of Electrical and Computer Engineering and holds joint appointments in the Departments of Radiology, Biomedical Engineering, and Mathematical Sciences. He is also associate director for Research of the Center for Integrated Surgical Systems and Technology, a National Science Foundation Engineering Research Center. He was an associate editor of *IEEE Transactions on Image Processing* from 1992-1995, and is currently associate editor of *IEEE Transactions on Medical Imaging*. Dr. Prince received a 1993 US National Science Foundation Presidential Faculty Fellows Award and was Maryland's 1997 Outstanding Young Engineer. His current research interests are in image processing and computer vision with primary application to medical imaging. Dr. Prince is a senior member of the IEEE and a member of Sigma Xi. He is also a member of the Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi honor societies.