A Novel Active Contour Framework: Multi-component Level Set Evolution under Topology Control

Florent Segonne, Jean-Philippe Pons,
Bruce Fischl and Eric Grimson

AI Memo 2005-020 June 2005

Abstract

We present a novel framework to exert a topology control over a level set evolution. Level set methods offer several advantages over parametric active contours, in particular automated topological changes. In some applications, where some a priori knowledge of the target topology is available, topological changes may not be desirable. A method, based on the concept of simple point borrowed from digital topology, was recently proposed to achieve a strict topology preservation during a level set evolution. However, topologically constrained evolutions often generate topological barriers that lead to large geometric inconsistencies. We introduce a topologically controlled level set framework that greatly alleviates this problem. Unlike existing work, our method allows connected components to merge, split or vanish under some specific conditions that ensure that no topological defects are generated. We demonstrate the strength of our method on a wide range of numerical experiments.

1. Introduction

Active contours constitute a general technique of matching a deformable model onto an image by means of energy minimization. Curves or surfaces deform within a 2-dimensional or 3-dimensional image subject to both internal and external forces and external constraints. Since their introduction by Kass et al. in [13], active contours have benefited many computer vision research areas (image segmentation [3], surface reconstruction [19, 4], shape modelling, visual tracking [18], and stereovision [5, 12, 7], etc.).

Depending on the implementation, there are essentially two ways of representing an active contour. Parametric approaches encode the manifold of interest with an explicit representation using a Lagrangian formulation, while geometric active contours are represented implicitly as level sets of functions defined on higher-dimensional manifolds in an Eulerian formulation [17]. Geometric active contours, which have been introduced by Caselles et al. [3], offer many advantages over parametric approaches. In addition to their ease of implementation, level sets do not require any parameterization of the evolving contour. Self-intersections, which are costly to prevent in parametric deformable models, are naturally avoided and topological changes are automated. Also, many fundamental properties of the active contours, such as the normal or the curvature, are easily computed from the level set function.

The ability to automatically change topology is often presented as an advantage of the level set method over explicit deformable models. However, this behavior is not desirable in some applications. This is typically the case in biomedical image segmentation, where the topology of the target shape is prescribed by anatomical knowledge. In order to overcome this problem, a topology preserving variant of the level set method has been proposed [10]. The level set function is iteratively updated with a modified procedure based on the concept of *simple point* from digital topology [1]; the final mesh is obtained with a modified topology-consistent marching cubes algorithm. This method ensures that the resulting mesh has the same topology as the user-defined initial level set.

While such topological preservation is desired in some applications, it is often too restrictive. Because the different components of the contour are not allowed to merge or to split up, the number of connected components will remain constant throughout the evolution. This number must be known by the user *a priori* and the initial contour must be designed accordingly. Also, the sensitivity to initial conditions, which already limits the applicability and efficiency of active contour methods, is considerably increased. The initial contour must both have the same topology as the target shape and be close enough to the final configuration, otherwise the evolution is likely to be trapped in topological dead-ends including large geometric inconsistencies (*cf* Fig.1 b&e).

While being able to control the topology of an active contour is an attractive feature, forcing it to remain identical through an evolution constitutes a very strong constraint. In this paper, we propose a method to exert a more subtle topological control on a level set evolution. Some *a priori* knowledge of the target topology can be integrated without requiring that the topology be known exactly. Our method greatly alleviates the sensitivity to initial conditions by allowing connected components to merge, split or vanish without introducing any topological defects (such as handles). For example, an initial contour with a spherical topology may split into several pieces, go through one or several mergings, and finally produce a certain number of contours, all of which will be topologically equivalent to a sphere. A subset of these components may then be selected by the user as the desired output (typically the largest component if one spherical contour is needed, the others being caused by noise).

Our approach is based on an extension of the concept of *simple point* to multi-label images, that we have called *multisimple point*. This criterion ensures that no topological defects are generated while splitting or merging the components of the object. This algorithm fills the gap between the original level set framework and topology preserving level set methods.

2. Background

2.1. Topology

Topology is a branch of mathematics that studies the properties of geometric figures that are preserved through deformations, twistings and stretchings, hence without regard to size, absolute position.

Here, we focus on closed contours (i.e. compact surfaces) in three dimensions. Any compact connected orientable surface is homeomorphic to a sphere with some number of handles. This number of handles is a topological invariant called the *genus*. For example, a sphere is of genus 0 and a torus is of genus 1. The genus g is directly related to another topological invariant called the *Euler characteristic* χ by the formula $\chi = 1 - 2g$. The Euler characteristic is of great practical interest because it can be calculated from any polyhedral decomposition of the surface by $\chi = V - E + F$, where V, E and F denote respectively the number of vertices, edges and faces of the polyhedron.

Homeomorphisms are used to define the *intrinsic* topology of an object, independently of the embedding space. For example, a knotted solid torus has the same genus as a simple torus, or a hollow sphere as two spheres. In order to topologically differentiate these surfaces, one needs a theory that considers the embedding space. Homotopy, which defines two surfaces to be homotopic if one can be continuously transformed into the other, is such a theory that provides a measure of an object's topology (see [11] for an excellent course in algebraic topology).

2.2. Digital topology

Digital topology provides an elegant framework, which transposes the continuous concepts of topology to discrete images. In this theory, binary images inherit a precise topological meaning. In particular, the concept of homotopic deformation, necessary to assign a topological type to a digital object, is clearly defined through the notion of *simple point*. An extensive discussion of these concepts can be found in the work of Mangin et al [16]. In this section, some basic notions of digital topology are presented. All definitions are from the work of G. Bertrand, which we refer to for more details [1].

A 3D binary digital image I is composed of a foreground object X and its inverse, the complement $\overline{X}$. We first need the concept of *connectivity*, which specifies the condition of adjacency that two points must fulfill to be regarded as connected. Three types of connectivity are commonly used in 3D: 6-, 18- and 26-connectivity. Two voxels are 6-adjacent if they share a face, 18-adjacent if they share at least an edge and 26-adjacent if they share at least a corner. In order to avoid topological paradoxes, different connectivities, n and $\overline{n}$, must be used for X and $\overline{X}$. This leaves us with four pairs of compatible connectivities: (6,26), (6,18), (26,6) and (18,6). In the following, we note $N_n(x)$ the n-neighborhood of a point x, i.e. the set of grid points which are n-adjacent to x. We also set $N_n^*(x) = N_n(x) \setminus \{x\}$.

An n-path π is a sequence (possible empty) of points $x_0...x_k$ with x_i adjacent to x_{i-1} for $i = 1, ..., k$. The path is elementary if all points in the sequence are different except possibly $x_0 = x_k$. An elementary n-path is simple if each point of π has, at most, two n-adjacent points in π. If $x_0 = x_k$, then π is closed. Finally, we note that any path from x_0 to x_k contains an elementary path and a simple path from x_0 to x_k.

Once the concepts of connectivity and path have been introduced, the notion of topological defect can be defined. Topological defects in a digital volume are consistuted of holes (i.e. handles) and cavities. While a cavity in an object X (resp. $\overline{X}$) is easily detected as a finite $\overline{n}$-connected (resp. n-connected) components of $\overline{X}$ (resp. X), the notion of hole is not simple to define. The presence of a hole in X (resp. $\overline{X}$) is detected whenever there is a closed n-path (resp. $\overline{n}$-path) in X (resp. $\overline{X}$) that cannot be deformed in X (resp. $\overline{X}$) to a simple point.

We then go to the definition of a *simple point*. This concept is central to the method of [10] and to our method. A point of a binary object is *simple* if it can be added or removed without changing the topology of both the object and its background, i.e. without changing the number of connected components, cavities and handles of both X and $\overline{X}$. A simple point is easily characterized by two *topological* numbers with respect to the digital object X and a consistent connectivity pair $(n, \overline{n})$. These numbers, denoted $T_n(x, X)$ and $T_{\overline{n}}(x, \overline{X})$, have been introduced by G. Bertrand in [1] as an elegant way to classify the topology type of a given voxel. The values of $T_n(x, X)$ and $T_{\overline{n}}(x, \overline{X})$ characterize isolated, interior and border points as well as different kinds of junctions. In particular, a point is simple if and only if $T_n(x, X) = T_{\overline{n}}(x, \overline{X}) = 1$. Their efficient computation, which only involves the 26-neighborhood, is described in [2].

2.3. Topology preserving level sets

Han et al. [10] have used the concept of simple point to design a topology preserving variant of the level set framework. The binary object of interest is the interior of the contour, i.e. the domain where the level set function is strictly negative: $X = \{x \mid \Phi(x) < 0\}$. The digital topology of X is preserved during the evolution by the means of a modified update procedure detailed in Algorithm 1, below. This approach prevents *digital* non-simple grid points from changing sign, therefore retaining the initial digital topology throughout the level set evolution.

For this method to be useful, it must be complemented with a topology-consistent isocontour extraction algorithm. Standard marching squares or marching cubes algorithm [15] do not generate topologically consistent tessellations. In order to alleviate this problem, Han et al. have designed a modified connectivity-consistent marching contour algorithm, by building a specialized case table for each type of digital topology. Using the topology preserving level set algorithm and the topology-consistent marching contour algorithm in conjunction, with the same digital topology $(n, \overline{n})$ throughout the process, guarantees that the output mesh is topologically equivalent to the user-defined initial level set.

Algorithm 1 Topology preserving level sets. Han et al. [10]

```
for all iterations do
  for all grid points do
    Compute the new value of the level set function
    if the sign does not change then
      Accept the new value
    else {sign change}
      Compute the topological numbers
      if the point is simple then
        Accept the new value
      else
        Discard the new value
        Set instead a small value of the adequate sign
```

3. Methods

The simple point condition is a very efficient way to detect topological changes during a level set evolution. However, in many applications, the topology preserving level set method of Han et al. is too restrictive.

The primary concern is topological defects such as cavities and handles, which are difficult to retrospectively correct [9, 6, 8, 14]. On the other hand, changes in the number of connected components during the evolution are less problematic. Different connected components are easily identified using standard region growing algorithms. A subset of them may be selected by the user as the final output, typically the largest one if a single component is needed, the others being imputable to noise in the input data.

We extend the concept of simple point to allow distinct connected components to merge and split while ensuring that no additional handle or cavity is generated in the object. For example, an initial contour with spherical topology may split into several pieces, go through one or several mergings, and finally produce a specific number of surfaces, all of which are topologically equivalent to a sphere.

3.1. From simple points to multisimple points

The different values of T_n and $T_{\overline{n}}$ characterize the topology type of a given point x, providing important information with regards to its connectivity to the object X. In particular, isolated points are characterized by $T_n = 0$ and $T_{\overline{n}} = 1$, while different junctions by $T_n > 1$ and $T_{\overline{n}} = 1$. This additional information can be used to carefully design a multi-label digital framework, which allows different connected components to split, merge or vanish under topology control.

A point is *multisimple* if it can be added or removed without changing the number of handles and cavities of the object. Contrary to the case of simple points, the addition of a multisimple point may merge several connected components, and its removal may split a component into several parts.

We note $C_n(x,X)$ the set of n-connected components of $X \setminus \{x\}$ that are n-adjacent to x. If the cardinality of this set is strictly greater than one, the addition or removal of x involves a merge or a split respectively. A point is multisimple if and only if it is simple relatively to each component of $C_n(x,X)$ taken separately, resulting in an elementary criterion using topological numbers. Formally, a point is said to be multisimple relatively to an object X if and only if:

$$T_{\overline{n}}(x,\overline{X}) = 1 \qquad (1)$$

$$\forall\, C \in C_n(x,X),\ T_n(x,C) = T_{\overline{n}}(x,\overline{C}) = 1\ . \qquad (2)$$

proof In order to verify that the concept of multisimple point does not introduce any topological defects in the volume, we need to show that the addition or deletion of a multisimple point x does not introduce any holes or cavities in any of the connected components $C \in C_n(x,X)$ or $\overline{X}$, where $X = \bigcup_i C_i \bigcup \{x\}$.

A simple point $x \in X$, characterized by $T_n(x,X) = T_{\overline{n}}(x,\overline{X}) = 1$, can be removed without changing the topology of the image. This implies a one-to-one correspondence between the connected components, the holes of X and $\overline{X}$ and the connected components, the holes of $X \setminus \{x\}$ and $\overline{X} \bigcup \{x\}$, the n-connectivity being used for X and the $\overline{n}$-connectivity being used for $\overline{X}$. Using this property, the proof can easily completed as follow.

Holes : since the connected components $C_i \in C_n(x,X)$ are not adjacent, any simple closed path in $X = \bigcup_i C_i \bigcup \{x\}$ is strictly contained in $C_j \bigcup \{x\}$ for one of the connected components C_j. The multisimple criterion, which ensures that the point x is simple relatively to the component C_j (see Eq.2), guarantees that the addition or deletion of x does not create any n-holes in the process.

Similarly, any simple closed $\overline{n}$-path in the background component $\overline{X}\bigcup\{x\}$ is contained in $\overline{C}$ for any of the components $C \in C_n(x, X)$. Therefore, Eq.2 ensures that no $\overline{n}$-holes are created in $\overline{X}$.

Cavities : the point x is simple relatively to each component $C \in C_n(x, X)$. As a consequence, its addition or deletion does not introduce any cavities in any of the components C or its complement $\overline{C}$. Finally, we note that no cavities other than the ones formed by the n-connected components of $C_n(x, X)$ are generated in $\overline{X}$. If this were the case, since the components of $C_n(x, X)$ are not adjacent, this would mean that the cavity would have been created in one of the components $C \in C_n(x, X)$, which would contradict what we have just proven.

When merging or splitting connected components by adding or removing a multisimple point, the total genus (i.e. the total number of handles) of the different components is preserved. For example, a torus and a sphere will merge into a torus. A double torus may split into two toruses or into a double torus and a sphere. We note that, under this condition, an isolated point is a multisimple point, which allows components to be created or to disappear.

3.2. Level set evolution under topology control

With the concept of multisimple point in hand, we are now ready to describe our new level set framework. Similarly to the approach of Han et al. [10], we exploit the binary nature of the level set function that partitions the underlying digital image into strictly negative inside points and positive outside points. During the evolution, we maintain a map L of labels coding for the different connected components of X. The background $\overline{X}$ is assigned the 0 label.

The update procedure for each grid point at each iteration is described in Algorithm 3. The simple point condition, more restrictive, is checked first, because it is computationally cheaper. If the point is non-simple, then $C_n(x, X)$ is computed in order to check the multisimple criterion.

Interestingly, if x is part of the background object and is a candidate for addition, $C_n(x, X)$ can be deduced directly from the map L. If x is a candidate for removal, the complete set $C_n(x, X)$ must be computed. However, when dealing with components that do not possess any handles, the most common situation in practice, the computation of $C_n(x, X)$ only involves local computations.

Algorithm 3 also describes how to consistently update the labels of the different connected components. The more complex case is the removal of a multisimple point involving a split: in this case, some unused labels must be assigned to the new connected components.

The resulting procedure is an efficient level set model that prevents handles and cavities from being created during the evolution, allowing the number of connected components to vary.

3.3. Extension

In this section, we show how our method can be modified to allow the generation of cavities. Unexpected cavities, which can be interpreted as background $\overline{n}$-connected components, might generate large geometrical inaccuracies in level set evolutions that prevent their formation. This might be of particular importance when dealing with noisy data or in medical imaging, when unexpected medical structures, such as tumors, might exist in the volume to be segmented.

Algorithm 2 Level sets under topology control with handle and cavity preservation

```
Compute the new value of the level set function
if the sign does not change then
    Accept the new value
else {sign change}
    Compute the topological numbers
    if the point is simple then
        Accept the new value
        if negative new value then
            $L(x) \leftarrow$ the only non-zero label $n$-adjacent to $x$
        else
            $L(x) \leftarrow 0$
    else {non-simple point}
        Compute $C_n(x, X)$
        Compute the topological numbers of each component
        if the point is multisimple then
            Accept the new value
            if negative new value then
                Fill all $C \in C_n(x, X)$ with the same label
            else
                Fill each $C \in C_n(x, X)$ with a different label
        else {non-multisimple point}
            Discard the new value
            Set instead a small value of the adequate sign
```

The concept of multisimple point evaluates if the addition or removal of a point will change the number of handles or cavities of an object. In particular, this allows a single component to split into several connected components, without introducing any topological defects. This splitting process can be interpreted as the generation of one or several background cavities. Therefore, the extension of the concept of multisimple point to both foreground and background components will naturally allow for the formation of cavities during the level set evolution. Formally, a point x is said to be multisimple relatively to X and $\overline{X}$ if and only if:

$$\begin{gathered} T_{\overline{n}}(x, \overline{X}) = 1 \\ \forall\ C \in C_n(x, X),\ T_n(x, C) = T_{\overline{n}}(x, \overline{C}) = 1 \\ \text{or} \\ T_n(x, X) = 1 \\ \forall\ C \in C_{\overline{n}}(x, \overline{X}),\ T_n(x, \overline{C}) = T_{\overline{n}}(x, C) = 1 \end{gathered}$$

In this modified level set evolution, background components need to be assigned distinct labels. The map L of labels encodes the different connected components of X and $\overline{X}$. For each grid point, the conditions for multisimple point are verified relatively to X and $\overline{X}$. Some care must be taken in order to ensure that the map of labels is correctly updated. The first equation of each condition guarantees that sign changes do not generate ambiguous label changes. For each point $x \in X$

changing sign, only one single connected component C of the inverse object $\overline{X}$ is adjacent to the point x; unambiguously, the newly assigned label of x is the one of the component C.

The resulting level set evolution allows for the generation of cavities, leading to a flexible model that only prevents the formation of handles. We insist on the fact that digital topology does not provide a consistent framework for multi-label images. However, by ensuring that no components of the same object X or $\overline{X}$ are adjacent, topological inconsistencies are avoided.

3.4. Implementation Issues

During a level set evolution, most points of the narrow-band do not change sign and do not imply a potential change of topology. The simple point condition, which only involves local calculations, is computationally cheap and leads to an efficient algorithm (see [2] and [10]).

Similarly, when the initial level set components do not possess any handles, multisimple points, which constitute a direct extension of the concept of simple points, only require local computations. This situation is the most common in practice, since one is most often interested in segmenting structures that have a spherical topology. However, we note that, when dealing with components that do possess handles (i.e. such as a torus), the computation of the set $C_n(x, X)$ might be expensive, as it involves finding the set of n-connected components of $X \setminus \{x\}$ that are n-adjacent to x.

The merging of connected components into one single component can easily be done using the label map L. If a point x is part of the background object and is a candidate for addition, $C_n(x, X)$ can be deduced directly from the map L. The merging step simply amounts to assigning the same label to each neighboring connected components.

One must be more careful when splitting components (i.e. split of a foreground component into several components or generation of cavities). During the evolution, some components may need to be split into several components, which requires the assignment of some unused labels to the new connected components that must be previously identified. This can be done using standard region growing algorithms.

Also, an update of the narrow band might generate a series of *useless and computationally expensive* splits depending on the ordering of the points in the narrow band. This would be the case, for instance, when a one-voxel wide region is iteratively broken into several pieces. In order to avoid the useless generation of connected components, we prevent components from splitting during the first pass in the narrow band by refusing sign changes of candidate voxels for a split. They are assigned a small value of the appropriate sign and an indicative flag is used to signal a potential component split. After the first pass, the multisimple condition is re-evaluated for each candidate voxels and the components are retrospectively updated. Finally, we note that, in order to avoid useless multiple splits of the same component into several pieces, every candidate voxel that is adjacent to an updated voxel has its flag deleted once a split has happened.

Algorithm 3 Level sets under topology control with handle preservation

```
Compute the new value of the level set function
if the sign does not change then
    Accept the new value
else {sign change}
    Compute the topological numbers
    if the point is simple then
        Accept the new value
        if negative new value then
            $L(x) \leftarrow$ the only foreground label $n$-adjacent to $x$
        else
            $L(x) \leftarrow$ the only background label $\overline{n}$-adjacent to $x$
    else {non-simple point}
        if { $T_{\overline{n}}(x, \overline{X}) \neq 1$ and $T_{\overline{n}}(x, \overline{X}) \neq 1$} then
            Discard the new value
            Set instead a small value of the adequate sign
        else {check if the point is multisimple}
            if $T_{\overline{n}}(x, \overline{X}) = 1$ then
                Compute $C_n(x, X)$
                Compute the topological numbers of each component $C \in C_n(x, X)$
                if the point is multisimple relatively to $X$ then
                    Accept the new value
                    if negative new value then
                        Foreground components' merge
                        Fill all $C \in C_n(x, X)$ with the same label
                        $L(x) \leftarrow$ the only foreground label $n$-adjacent to $x$
                    else
                        Foreground component split or deletion
                        Fill each $C \in C_n(x, X)$ with a different label
                        $L(x) \leftarrow$ the only background label $\overline{n}$-adjacent to $x$
                else {non-multisimple point}
                    Discard the new value
                    Set instead a small value of the adequate sign
            else {$T_n(x, X) = 1$}
                Compute $C_{\overline{n}}(x, \overline{X})$
                Compute the topological numbers of each component $C \in C_{\overline{n}}(x, \overline{X})$
                if the point is multisimple relatively to $\overline{X}$ then
                    Accept the new value
                    if negative new value then
                        Cavity formation or deletion
                        Fill all $C \in C_{\overline{n}}(x, \overline{X})$ with different labels
                        $L(x) \leftarrow$ the only foreground label $n$-adjacent to $x$
                    else
                        Merge of background components
                        Fill each $C \in C_{\overline{n}}(x, \overline{X})$ with the same label
                        $L(x) \leftarrow$ the only background label $\overline{n}$-adjacent to $x$
                else {non-multisimple point}
                    Discard the new value
                    Set instead a small value of the adequate sign
```

4. Experiments and Applications

In this section, we present some experiments illustrating the performance of the proposed approach and the panel of potential applications. In the following examples, we use the extended version of our level set framework.

4.1. ’C’ shape

’C’ shape First, we consider the segmentation of a simple ’C’ shape under two different initializations (*cf* Fig.1 *a-c* & *d-f*). Our method, columns *c*,*f*, is compared to the original level set formulation, columns *a*,*d*, and the topology preserving model introduced by Han et al. [10], columns *b*,*e*. In the

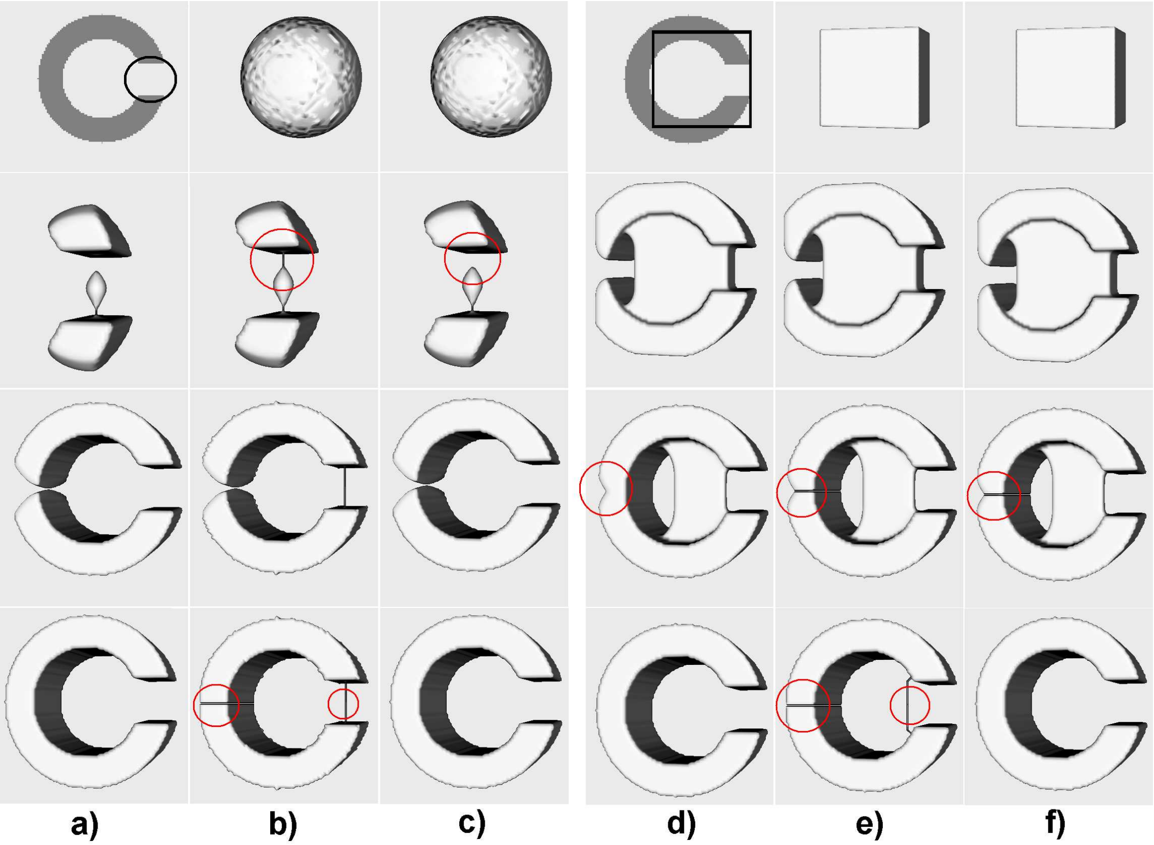

Fig. 1. Segmentation of a ’C’ shape using a spherical initialization a-c and a rectangular initialization d-f. The top row shows cuts of the 3D shape locating the initial component. a,d: Traditional level set evolution. b,e: Topology preserving level set evolution. c,f: level set evolution under topology control. Differences of behavior are circled in the images. Note that in both cases, our level set framework is able to correctly segment the shape of interest, without generating invalid topology during the evolution.

case of this simple example, standard level sets produce accurate segmentations. However, they do not provide control over the final topology, as illustrated by the second initialization, column *d*. During the evolution, the level set automatically changes topology, temporarily generating a toroïdal shape.

On the other hand, topology preserving level sets, while guaranteeing the correct topology, might easily be geometrically inaccurate (see Fig.1*b*,*e*). Topological barriers, generated during the evolution to prevent topological changes, generate incorrect contour configurations which are difficult to retrospectively correct. We note that even close initializations (*cf* column *e*) can lead to inaccurate segmentations.

The behavior of our approach corresponds to a trade-off in between the previous two methods. Our level set framework, which offers more flexibility during the evolution, greatly alleviates the sensitivity to initialization. It provides a control over the topology of all connected components, without enforcing strong topological constraints. The first initialization (*cf* column *c*) produces a level set evolution, which splits the active contour into three distinct components with a spherical topology. While one of these components vanishes, the two other components merge, closing the 'C' shape. In this case, topologically controlled level sets behaved in exactly the same way traditional level sets do. Our method is still sensitive to topological barriers, in the sense that it won't allow component to generate handles. However, it is less likely to be trapped into topological dead-ends than a strict topology preserving model. This illustrated in Fig.1 *f*, where the spherical component is prevented from generating a handle until it gets separated into two distinct components.

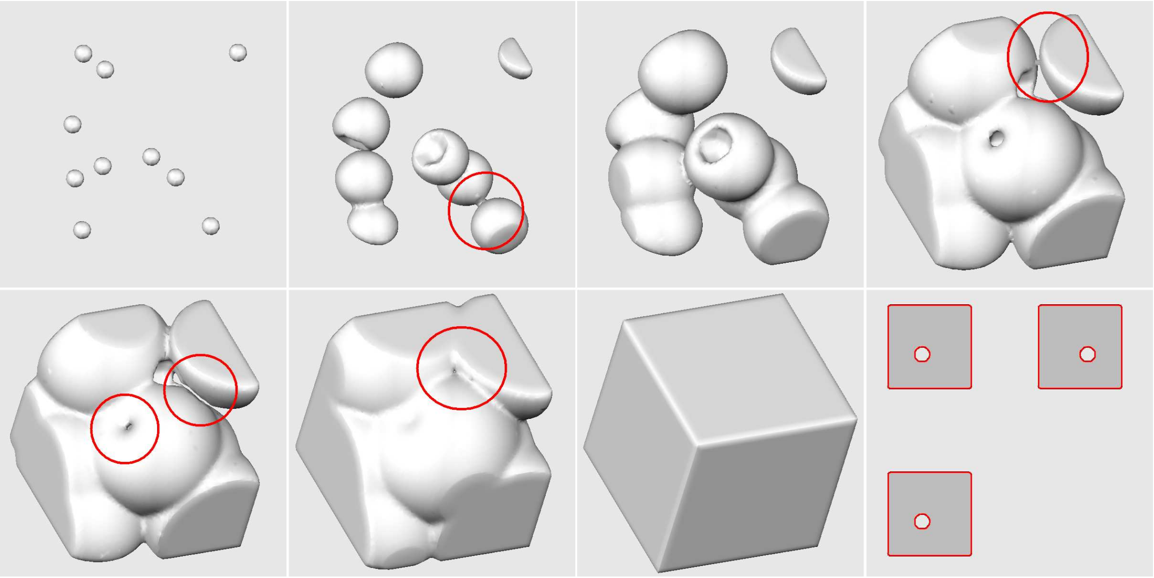

Fig. 2. Segmentation of a cubic shape containing 3 cavities. 10 initial seed points were randomly selected. Note how components split, merge and disappear during the evolution, and how the active contour encloses cavities.

4.2. Formation of cavities

The second experiment, see Fig.2, illustrates the ability of our approach to generate cavities during the evolution. The object to be segmented is a synthetic cube, containing 3 large cavities. 10 seed points, randomly selected inside and outside the volume, were used to initialize the level set evolution reported in Fig.2. During the evolution, components merge, vanish and produce cavities, generating a final accurate representation constituted of 3 spherical cavities and the main object. We note that all the components are easily extracted, since they carry distinct labels that are iteratively updated during the evolution.

4.3. Cortical segmentation

In a third experiment, we apply a naïve approach to the difficult problem of cortical surface extraction to illustrate the strength of our method. Excluding pathological cases, the cortex, which corresponds to a highly-folded thin sheet of gray matter, has the topology of a sphere. The extraction of accurate and topologically correct cortical representations is still an active research area [6, 9]. In this example, the cortical surface is initialized with 55 spherical components, automatically selected in a greedy manner, such that every selected point is located at a minimal distance of $10mm$ from the previous ones (see Fig.3). During the evolution, the components progressively merge together and enclose cavities, resulting in a final surface composed of one single spherical component with 5 cavities. The same evolution without topology constraint produces a final cortical surface with 18 topological defects (i.e. handles).

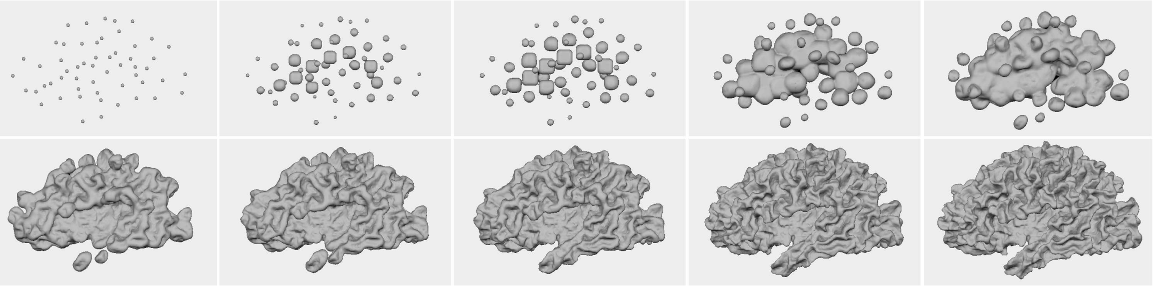

Fig. 3. Segmentation of the cortex from an anatomical MRI. The initial level set was constituted of 55 connected components. The final surface has a spherical topology, corresponding to an Euler number of 2. The same level set evolution without topological control results in a final surface with 18 topological defects (Euler number of $\chi = -34$)

.

4.4. Segmentation of blood vessels

Finally, we show how our method could be applied to the segmentation of blood vessels from Magnetic Resonance Angiography. Because these vessels do not split and merge, their topology is

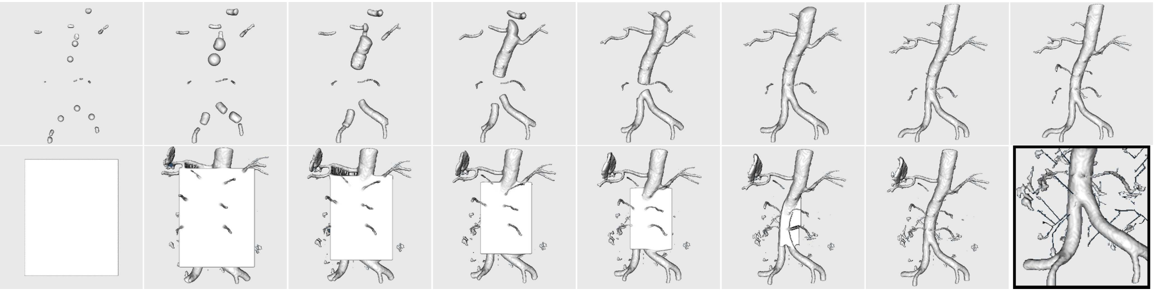

Fig. 4. Segmentations of blood vessels in a 3D angiography under two different initializations. Top row: 20 seed points were selected to initialize the active contour, which generates 3 components. Bottom row: An enclosing contour is used to initialize the level set. After 9 mergings and 99 splittings, the final segmentation is constituted of 91 components, 53 of which were due to random noise. The last image in the black frame shows the final segmentation produced by a topologically constrained evolution under the same initialization.

the one of several distinct components with no handles (i.e. each component has the topology of a sphere). While traditional level sets produce segmentations that could include topological defects, topologically constrained level sets would result in a slow and laborious segmentation. Since the simultaneous evolution of several components, which cannot be merged together, can easily be trapped in topological dead-ends, each component would need to be iteratively initialized, when the evolution of the previous one has terminated.

On the other hand, our method offers the possibility to concurrently evolve multiple components that can merge, split and vanish. The initial contour can be initialized by a set of seed points, manually or automatically selected, or by a single enclosing component, without affecting much the final representation.

Figure 4 shows the segmentation of an angiography under two different initializations. In a first experiment (*cf* top row), 20 seed points were automatically selected at the brightest locations in the MRA. The level set evolution iteratively merged most of the components, generating a final segmentation with 3 spherical components. In the second experiment (*cf* bottom row), one single global component, enclosing most of the object of interest, was used to initialize the active contour. During the evolution, 9 components merged and 99 split producing a final segmentation composed of 91 components, 53 of which were single voxel components due to random noise in the imaging process.

5. Discussion and Conclusion

We introduce a new level set framework that offers control over the topology of the level set components during the evolution. Contrarily to previous approaches that either do not constrain the topology or enforce a hard topological constraint, our method exerts a subtle control over the topology of each component to prevent the formation of topological defects, such as handles (or cavities depending on the application and the choice of active contour model). Distinct components can merge, split or disappear during the evolution, but no handles (or cavities) will be generated. In particular, a contour solely composed of spherical components will only produce spherical components throughout the evolution. In this case, which is the most common situation in practice, all computations are local and the multisimple point checking can be done efficiently. The only computational complexity comes from the generation of new components, as new labels need to be assigned to each of them.

While the original level set model does not provide any topological control, topology preserving level sets enforce a strong constrain that is often too restrictive. Our framework establishes a tradeoff in between the two models. It offers a subtle topological control that alleviates most problems of methods that enforce a strong topological constrain (i.e. sensitivity to initialization and noise, simultaneous evolution of multiple components and speed of convergence). The experiments presented in this paper illustrate some potential applications that could greatly benefit from our approach.

References

1. G. Bertrand. Simple points, topological numbers and geodesic neighborhoods in cubic grids. *Pattern Recognition Letters*, 15(10):1003–1011, 1994.
2. G. Bertrand. A boolean characterization of three-dimensional simple points," pattern recognition letters. *Pattern recognition letters*, 17:115–124, 1996.
3. V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *The International Journal of Computer Vision*, 22(1):61–79, 1997.
4. Y. Duan, L. Yang, H. Qin, and D. Samaras. Shape reconstruction from 3D and 2D data using PDE-based deformable surfaces. In *European Conference on Computer Vision*, volume 3, pages 238–251, 2004.
5. O. Faugeras and R. Keriven. Variational principles, surface evolution, PDE's, level set methods and the stereo problem. *IEEE Transactions on Image Processing*, 7(3):336–344, 1998.
6. B. Fischl, A. Liu, and A.M. Dale. Automated manifold surgery: Constructing geometrically accurate and topologically correct models of the human cerebral cortex. *IEEE Transaction on Medical Imaging*, 20:70–80, 2001.
7. B. Goldlücke and M. Magnor. Space-time isosurface evolution for temporally coherent 3D reconstruction. In *International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 350–355, 2004.
8. I. Guskov and Z. Wood. Topological noise removal. *Graphics Interface Proceedings*, pages 19–26, 2001.

9. X. Han, C. Xu, U. Braga-Neto, and J.L. Prince. Topology correction in brain cortex segmentation using a multiscale, graph-based approach. *IEEE Transaction on Medical Imaging*, 21(2):109–121, 2002.
10. X. Han, C. Xu, and J.L. Prince. A topology preserving level set method for geometric deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):755–768, 2003.
11. Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.
12. H. Jin, S. Soatto, and A.J. Yezzi. Multi-view stereo beyond Lambert. In *International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 171–178, 2003.
13. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *First International Conference on Computer Vision*, pages 259–268, London, June 1987.
14. N. Kriegeskorte and R. Goeble. An efficient algorithm for topologically segmentation of the cortical sheet in anatomical mr volumes. *NeuroImage*, 14:329–346, 2001.
15. William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In M.C. Stone, editor, *Proceedings of the SIGGRAPH*, pages 163–169, Anaheim, CA, July 1987. in Computer Graphics, Volume 21, Number 4.
16. 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. *Journal of Mathematical Imaging and Vision*, 5:297–318, 1995.
17. S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.
18. N. Paragios and R. Deriche. Geodesic active regions and level set methods for motion estimation and tracking. *Computer Vision and Image Understanding*, 97(3):259–282, 2005.
19. H. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and non-parametric shape reconstruction from unorganized points using a variational level set method. *Computer Vision and Image Understanding*, 80(3):295–314, 2000.