To appear in the *2000 IEEE Conf. on Computer Vision and Pattern Recognition*

# Maintaining Valid Topology with Active Contours: Theory and Application

Amitha Perera[1] Chia-Ling Tsai[1] Robin Y. Flatland[2] Charles V. Stewart[1]
perera@cs.rpi.edu tsaic@cs.rpi.edu rflatland@siena.edu stewart@cs.rpi.edu
[1] Rensselaer Polytechnic Institute, Troy, NY 12180
[2] Siena College, Loudonville, NY 12211

**Abstract**

*We develop and prove correct an algorithm that enables active contours to correctly represent regions that undergo topology changes as the contours evolve. Using the incremental motion typical of active contours, we introduce the concept of* motion regions *to determine the new topology. When the topology changes (e.g. by contours intersecting), the motion regions are used to delete and and reconnect the contours to accurately describe the new region. Contour intersections can also occur without topology changes. These are also appropriately handled. The algorithm to perform this task is proved correct in a general framework that makes few assumptions about the contour representation. We describe how this algorithm is applied to a polygonal representation of the contours, and argue that it does not significantly affect execution time. Finally, this polygonal implementation is used in surface extraction and phytoplankton classification.*

## 1 Introduction

Since their introduction [5], active contour techniques have been used in many applications, including segmentation [3, 13], medical image analysis [2, 5, 6, 10], and tracking [1, 4]. The fundamental advantage of active contours is that constraints derived from image measurements and constraints derived from contour shape considerations may be combined in a single energy functional, which is minimized with respect to parameters describing the contour location. Contour connectivity is automatically enforced.

Initially, active contour techniques used fixed parameterizations, with energy penalties for size changes. With the addition of dynamic reparameterizations, expansion/contraction forces, and region-based energy terms ("active regions"), closed contours initialized in small regions can expand dramatically to describe regions of arbitrary shapes and sizes.

New problems arise when active contours are allowed to grow arbitrarily beyond an initial shape and size: contour self-intersections and changes in region topology. Topologically, the simplest form of a closed contour is a single, non-intersecting cycle, which encloses a region homeomorphic to a disk. As a contour expands, however, it may eventually wrap around an image structure, with opposite sides of the contour approaching each other and eventually intersecting. Figure 1 illustrates this. Such intersections indicate an inconsistency in the boundary representation. In many cases, though not all, these intersections indicate a change in region topology. In the example shown, the contour should be split into two cycles, and the region should become homeomorphic to a ring. Handling problems associated with self-intersections and topology changes is crucial to expanding the general utility of active region and closed active contour techniques.

Several methods have been introduced to address the self-intersection and topology problems. Some have artificially avoided self-intersections, either explicitly by detecting and preventing motions that cause an intersection, or implicitly by adding a repulsive force between segments of the contour [3]. These methods disallow changes in region topology, which means topology is fixed *a priori* and information available to dynamically discover the topology through contour evolution is suppressed.

Two current methods do allow topology changes. One represents each boundary contour as a cycle of polygonal edges whose vertices are restricted to the edges of the cells in a simplical decomposition of the image plane [9, 10]. When boundary contours intersect, the underlying cells are used to determine the new topology and to correct the polygonal representation as necessary. There are three drawbacks to this approach. First, the method assumes that contour intersections occur at two points, even though it is possible for intersections to occur at a single point (Fig-

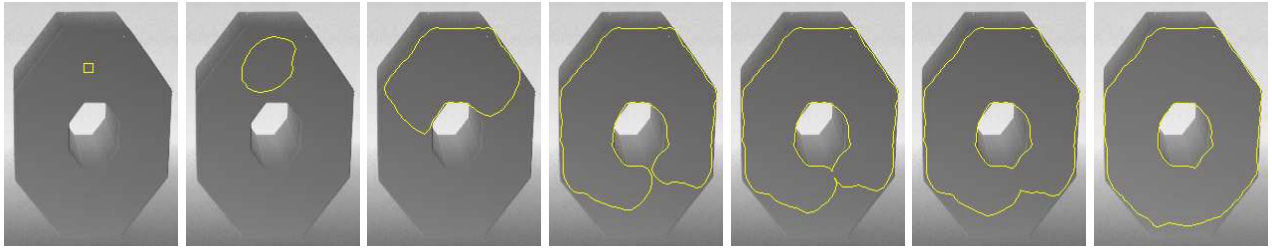


**Figure 1:** A result from surface extraction (Section 4). The contour is initialized to form a seed region. The contour, and hence the region, grows to cover the whole face of the nut. At a certain point, the contour self-intersects due to a topology change. The contour should then be split into two contours, after which region growth should continue to convergence.

ure 2(f)). Second, the underlying grid representation fundamentally changes the active contour implementation. It would be preferable to have a technique that could "plug-in" to other methods. Third, the boundary polygons are constrained to lie on a grid. The resolution of the representation is therefore limited by the resolution of the grid. The other method that allows topology changes represents the boundary of a region as the level set of an higher-order function [7, 12]. Here the topology of the level set can change without the topology of the higher-order function changing. This technique, however, requires careful formulation of a speed function governing the boundary's motion, maintains an implicit boundary representation which is less intuitive to manipulate than an explicit representation, and is grid based, thus incurring the two former two disadvantages above. A more detailed comparison of these techniques and the one presented here will be provided in a long version of this paper.

This paper presents a new, provably-correct technique to fix contour intersections and detect topology changes. The technique assumes a general boundary representation, which is simply a set of boundary contours (not necessarily polygons), and it places only a few constraints on what types of contour motions can and cannot occur. The constraints are easily met in practice, and do not detract from its generality and power. The technique is simple, computationally efficient, and easily incorporated into current implementations. One implementation is demonstrated using polygonal boundary contours. This is used in surface extraction and detection of phytoplankton in phase contrast microscopy images, demonstrating the power of using an active contour technique with self-correcting contours and topology.

## 2 Solution Overview

Active contours and active regions change shape in response to the gradient of an energy functional. This gradient is a function of the contour parameters, and hence, all the activity is along the boundary. Topology, by contrast, is a property of a region. From a topological viewpoint, curves are only used as a convenient representation of the region boundary and, by extension, of the region itself. Solving the contour intersection and topology problems requires a shift in thinking to focus on the region itself. As a result, our technique for correcting boundary contour intersections and for detecting topological changes depends on reasoning about region properties. While minimizing the energy functional is treated as primary, the changes in the boundary contour induced by the minimization are best viewed as changes to the region. Boundary contours in conflict with changes to the region interior or exterior are detected and deleted. This reflects the subservient role of the contours.

The technique described below is inductive in nature. It starts from a topologically correct, non-intersecting boundary representation. It incrementally moves the boundaries in minimizing the energy functional, detecting and correcting errors in the boundary representation. Nothing is assumed about the energy functional or the smoothness of the boundary contour. Therefore, some errors the algorithm detects and corrects result from motions that are unlikely to occur for some energy functionals. Importantly, however, the technique is made no more complicated by addressing these problems and therefore there is no cost to the generality of the algorithm.

Most incremental changes involve either the addition of new areas to the region or subtraction of current areas (Figures 2(a) and 2(b)). Some involve both (Figure 2(c)). Occasionally, contour intersections cause changes in topology (Figures 2(d) and 2(e)), but not always (Figure 2(f)). In addition, changes in topology are not always accompanied by contour intersections as shown in Figure 2(g). Here, the hole boundary passes completely through itself, eliminating the hole.

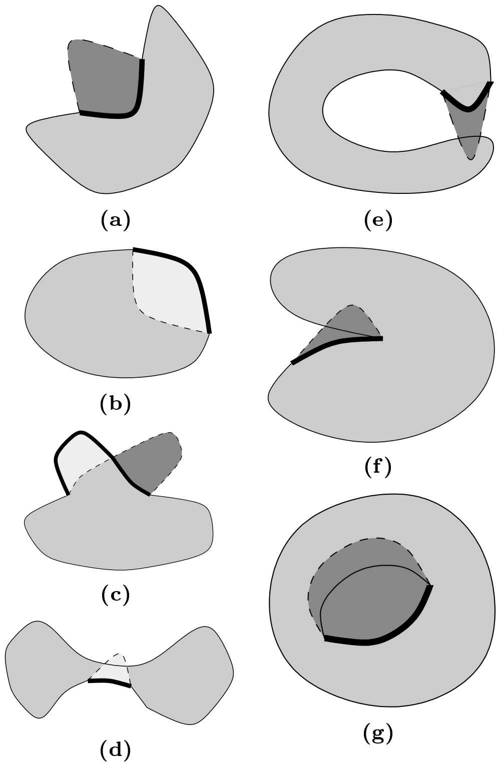


**Figure 2:** These figures illustrate the different types of boundary movements that could occur. The bold solid line and the dashed line give the old and new positions (respectively) of the moving curve portion. The thin solid lines are boundary curve portions that have not moved. The dark grey indicates a region that should be added to the current region, while the light grey should be subtracted. The medium grey is part of the current region unaffected by the motion.

As these examples show, each time a boundary moves there are two problems to be addressed. The first problem is determining if any boundary contours intersect and if so, removing the intersection in a way that is consistent with the new region. The second problem is detecting if a change in topology has occurred and if so, determining the new topology and taking appropriate actions. Detecting topology changes is particularly important when there are topological constraints placed on the regions (e.g. the region must have a face topology [8]).

The following three sections address the first problem. The second problem is closely tied to determining connectivity and is examined in Section 2.4.

### 2.1 Notation

All the regions and curves discussed in this paper are subsets of $\Re^2$. We assume the standard topology on $\Re^2$ [?].

Let $S \subset \Re^2$. Then $\overline{S}$ is the closure of $S$, $\mathring{S}$ is the interior of $S$, and $S^c$ is the complement of $S$ in $\Re^2$. A neighborhood of a point $x$ is an open set containing $x$. A point $x$ is a limit point of a set $S$ if every neighborhood of $x$ has a point other than $x$ in $S$. The boundary of a set $S$ is denoted $\delta S$ and can be defined by

$$\begin{aligned} \delta S \quad := \quad & \{x \in \overline{S} : \forall \text{ nbhoods } N \text{ of } x, \\ & N \cap S \neq \emptyset \text{ and } N \cap S^c \neq \emptyset\}. \end{aligned} \tag{1}$$

Clearly, a closed set contains its boundary while an open set does not.

A curve $\mathbf{c}$ in $\Re^2$ is a continuous map $\mathbf{c} : [0,1] \to \Re^2$. Abusing notation, we will also use $\mathbf{c}$ to denote the set $\{\mathbf{c}(t) : t \in [0,1]\}$ (which is the image of $\mathbf{c}$). A curve $\mathbf{c}$ is closed if $\mathbf{c}(0) = \mathbf{c}(1)$. Otherwise, it is open and $\mathbf{c}(0)$ and $\mathbf{c}(1)$ are its end-points. A closed curve has no end-points. A curve $\mathbf{c}$ is simple if it does not self-intersect. Boundary curves are oriented so that the interior of the region lies to the left, where "the left" of a curve $\mathbf{c}$ at a differentiable point $\mathbf{c}(t_0)$ is the left of the tangent direction at that point. The left at a non-differentiable point $\mathbf{c}(t_1)$ is defined in terms of the left of the points immediately before and after $\mathbf{c}(t_1)$.

A portion of a curve $\mathbf{c}(t)$, $t \in [0,1]$, is $\mathbf{c}(s)$ with $s \in [a,b]$, where $0 \leq a < b \leq 1$. Clearly a portion of a curve is itself a curve, and can be reparameterized such that the parameter is in $[0,1]$.

Two curves $\mathbf{c}$ and $\mathbf{d}$ (not necessarily distinct) *touch* at a point $x = \mathbf{c}(i) = \mathbf{d}(j)$ if there exists a line through $x$ separating $\mathbf{c}$ and $\mathbf{d}$ in a neighborhood $N$ of $x$. That is, there exists a line through $x$ such that $\mathbf{c} \cap N$ lies on or to the left of the line and $\mathbf{d} \cap N$ lies on or to the right of the line. If no such line exists, then the curves *cross* at $x$, and $x$ is called a crossing point on $\mathbf{c}$ and on $\mathbf{d}$. Note that under this definition, two curves overlapping are considered to be touching at each point of overlap.

A curve $\mathbf{c}(t)$ is oriented by its parameter $t$: from a point on the curve, we can "travel" backward (decreasing $t$) or forward (increasing $t$). Starting at a crossing point $x$ on a curve, the *next* crossing point is found by traveling forward from $x$ and the *previous* crossing point is found by traveling backward.

### 2.2 A correct boundary representation

We now define how the motion of the boundary contours changes the region. Define a region $\mathcal{R}$ as the closure of a bounded, open subset of $\Re^2$. The region will be represented by its boundary $\delta\mathcal{R}$. From the nature of $\mathcal{R}$, it follows that $\delta\mathcal{R}$ is a set of closed curves $c_i$. Without loss of generality (WLOG), assume each curve is parameterized over $[0,1]$. Clearly, the boundary curves do not cross. We impose a further condition

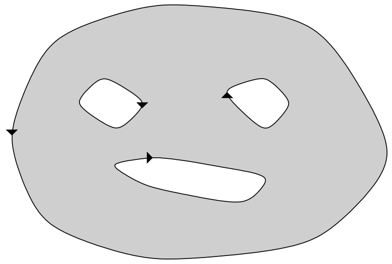


**Figure 3:** The shaded area $\mathcal{R}$ is the closure of a connected, open subset of $\Re^2$. The solid lines form the set of boundary curves $\delta\mathcal{R}$. The boundary curves are oriented such that the interior of the region lies to the left of each curve.

on $\mathcal{R}$ that its boundary curves do not touch. While not strictly necessary, this condition is easily met in practice and simplifies much of the discussion below. The boundary curves are assumed to be oriented such that $\mathcal{R}$ lies on the left of each curve (Figure 3). Abusing notation, $\delta\mathcal{R}$ is used to refer to the boundary both as a set of points and as a set of curves.

Formally, movement of the boundaries occurs when a portion $\mathbf{c}_m$ of a boundary curve is modified (moved) to become $\mathbf{c}'_m$. In the most common case, illustrated in Figures 4(a) and 4(b), $\mathbf{c}_m$ and $\mathbf{c}'_m$ define a single region which we call the *motion region*. The motion region could be additive, causing the region to grow (Figure 4(a)), or it could be subtractive, causing the region to shrink (Figure 4(b)). Sometimes, the motion of a single curve portion can result in multiple regions (Figure 4(c)). This case can be regarded as the simultaneous movement of two curve portions resulting in two distinct motion regions, one additive and one subtractive.

When arbitrary movement is permitted, more complicated cases can arise where the effect of the motion is not well-defined. In Figure 4(d) for example, it is not clear whether the striped region should or should not be part of the new region. For this reason, one constraint that we place on the motion is that the motion regions are well-defined. A second constraint is that the additive and subtractive regions are disjoint—if they were not, then it is not clear if the points in their intersection belong to the new region. For a motion region defined by $\mathbf{c}_m$ and $\mathbf{c}'_m$ to satisfy the first constraint, it is sufficient that (1) $\mathbf{c}'_m$ has the same endpoints as $\mathbf{c}_m$; (2) $\mathbf{c}'_m$ does not self-intersect; (3) $\mathbf{c}'_m$ does not intersect $\mathbf{c}_m$ except at the end points; and (4) $\mathbf{c}'_m$ does not touch $\delta\mathcal{R}$ (they may cross, however, as illustrated in Figures 2(e) and 2(d)).

Constraint (3) avoids the case illustrated in Figure 4(d). It does not prevent the case illustrated in Figure 4(c) because, as mentioned above, this case can be reformulated as the simultaneous movement of two independent curve portions. Constraint (4) maintains the restriction that the boundary curves do not touch.

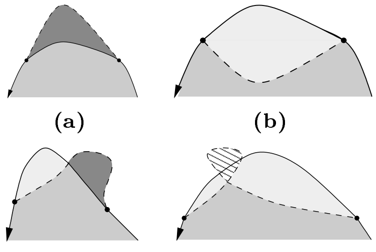



**Figure 4:** In all the diagrams above, the dark grey shows an additive motion region and the light grey a subtractive motion region. The medium grey area is part of $\mathcal{R}$. The dashed line is $\mathbf{c}'_m$ while the solid line between the two dots is $\mathbf{c}_m$.

These conditions are sufficient to make the motion regions well defined, but they are not necessary. Any set of conditions that clearly define the motion regions and ensure that the additive and subtractive regions do not intersect will also be sufficient. We present the four conditions above because they can be easily satisfied by any polygonal implementation that moves a single vertex at a time, and because polygonal boundary representations are widely used in active contour implementations.

**Definition 1.** Let $\mathbf{c}_m$ and $\mathbf{c}'_m$ be as defined above. The motion region $M$ is defined to be the open region enclosed by $\mathbf{c}_m$ and $\mathbf{c}'_m$. If $M$ lies to the left of $\mathbf{c}'_m$, then it is called an additive motion region. Otherwise, $M$ is called a subtractive motion region.

Joining all the additive motion regions into a region $M_a$ and all the subtractive motion regions into $M_s$, we can define

$$\mathcal{R}_{new} := \overline{(\mathcal{R} \cup \overline{M_a}) - \overline{M_s}}. \tag{2}$$

$\mathcal{R}_{new}$ is the new region that results from the motion. The goal of a topology correction algorithm is to determine its boundary $\delta\mathcal{R}_{new}$.

Notice that portions of an additive region may actually overlap $\mathcal{R}$, as seen in Figure 2(e). It can be shown that $\mathcal{R}_{new}$ is the closure of an open, bounded subset of $\Re^2$, and its (true) boundary, $\delta\mathcal{R}_{new}$, is a set of disjoint simple closed curves. However, $\mathcal{R}_{new}$ is not necessarily connected (even if $\mathcal{R}$ is), and in rare cases it could even be empty!

Let $\delta'\mathcal{R}$ be $\delta\mathcal{R}$ with the curve portions $\mathbf{c}_m$ replaced with the new curves $\mathbf{c}'_m$. The problem of correcting the contours can now be restated as the problem of matching $\delta'\mathcal{R}$ to $\delta\mathcal{R}_{new}$. Let $\delta\mathcal{R}_c$ be $\delta\mathcal{R}$ without the $\mathbf{c}_m$, that is, let $\delta\mathcal{R}_c$ be the unmoving part of the boundary.

It is clear from the definition of the new region that $\delta\mathcal{R}$ agrees with $\delta\mathcal{R}_{new}$ everywhere except in the motion regions and their boundaries. Since $\delta\mathcal{R}$ only differs from $\delta'\mathcal{R}$ in the motion region boundaries, $\delta'\mathcal{R}$ also agrees with $\delta\mathcal{R}_{new}$ everywhere except in the motion regions and their boundaries, and thus we can focus our attention on the motion regions and their boundaries. Further, $\delta\mathcal{R}_{new} \subseteq \delta'\mathcal{R}$, and thus the problem simplifies to determining which portions of $\delta'\mathcal{R}$ must be deleted.

In the next section, we will show that the portions to be deleted can be determined by examining the intersections of the curves in $\delta'\mathcal{R}$. Moreover, the examination will only use information local to the intersection point, using the motion region to (implicitly) translate the local information into global information.

### 2.3 Correcting the contours

Let $\mathbf{c}(s)$ and $\mathbf{d}(t)$ be a pair of closed boundary curves (not necessarily distinct) in $\delta'\mathcal{R}$ that cross at $x = \mathbf{c}(s_1) = \mathbf{d}(t_1)$. Since the motion regions are disjoint and the curves in $\delta\mathcal{R}$ do not cross, exactly one of $\mathbf{c}$ and $\mathbf{d}$ must belong to the boundary of a motion region $M$ in a neighborhood of $x$. WLOG, assume that $\mathbf{c} \subset \delta M$ in the neighborhood, which means that $\mathbf{d}$ crosses from $M$ into $M^c$ (or the other way) in the same neighborhood (Figure 5). Let $(s_0, s_2) \subset \Re$ with $s_1 \in (s_0, s_2)$ such that

$$\big(\forall s \in (s_0, s_2)\big)\big(\mathbf{c}(s) \in \delta M\big).$$

Similarly, let $(t_0, t_2) \subset \Re$ with $t_1 \in (t_0, t_2)$ such that either

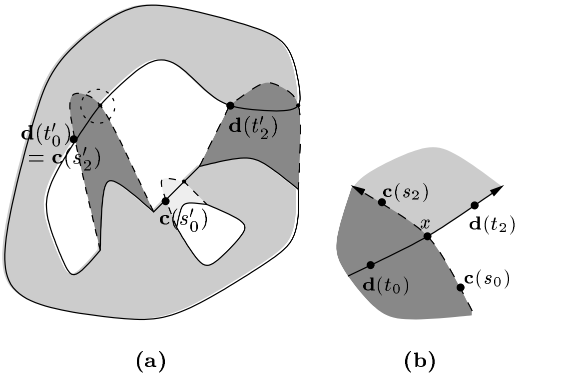
$$\mathbf{d}\big((t_0, t_1)\big) \in M^c \text{ and } \mathbf{d}\big((t_1, t_2)\big) \in M,$$

where $\mathbf{d}(S) := \{\mathbf{d}(s) : s \in S\}$, or

$$\mathbf{d}\big((t_0, t_1)\big) \in M \text{ and } \mathbf{d}\big((t_1, t_2)\big) \in M^c.$$

Let $\mathbf{c}(s_0')$ be the previous crossing point on $\mathbf{c}$ before $x$ and $\mathbf{c}(s_2')$ be the next crossing point on $\mathbf{c}$ after $x$, and similarly for $\mathbf{d}(t_0')$ and $\mathbf{d}(t_2')$. Note that it is possible (and quite common in practice) that $\mathbf{c}(s_0') = \mathbf{c}(s_1)$ or $\mathbf{c}(s_1) = \mathbf{c}(s_2')$, and even more that $\mathbf{c}(s_0') = \mathbf{c}(s_1) = \mathbf{c}(s_2')$. Similarly for $\mathbf{d}$.

**Directive 1.** If $\mathbf{d}(t_0) \in M$, then delete the points before $\mathbf{d}(t_1)$ ending at $\mathbf{d}(t_0')$ and the points after $\mathbf{c}(s_1)$ ending at $\mathbf{c}(s_2')$, excluding the end-points $\mathbf{d}(t_1)$, $\mathbf{d}(t_0')$, $\mathbf{c}(s_0)$, and $\mathbf{c}(s_2')$. Otherwise, delete the points after $\mathbf{d}(t_1)$ ending at $\mathbf{d}(t_2')$ and the points before $\mathbf{c}(s_1)$ ending at $\mathbf{c}(s_0')$, again excluding the end-points themselves.




**(a)** **(b)**

**Figure 5:** (a) shows regions undergoing movement. (b) shows a close-up view of the dotted circle in (a). In both figures, $\mathbf{c}(s_1) = \mathbf{d}(t_1) = x$. See Section 2.3 for details.

Note that deleting portions of $\mathbf{c}$ and $\mathbf{d}$ means that the remainder of $\mathbf{c}$ and $\mathbf{d}$ must be spliced at $x$. The effect of applying Directive 1 to all the intersections is, in essence, to separate $\delta'\mathcal{R}$ into disjoint simple closed curves and to delete those completely inside or completely outside $\mathcal{R}_{new}$. Note that this is done with *local* knowledge only.

**Lemma 1.** *No point deleted by Directive 1 is in* $\delta\mathcal{R}_{new}$*.*

*Proof.* There are four cases to examine: $M$ could be an additive or subtractive region, and $\mathbf{d}(t_0)$ could be in $M$ or in $M^c$.

Take the case of an additive motion region $M$ with $\mathbf{d}(t_0) \in M$ (Figure 5(b)). Since $\mathbf{d}(t_0')$ is the first point before $\mathbf{d}(t_1)$ to intersect $\delta M$, and since the boundary curves do not touch, it follows that $\big(\forall t \in (t_0', t_1)\big)\big(\mathbf{d}(t) \in M\big)$. Then, for each of these $\mathbf{d}(t)$, there exists a neighborhood $N \subset M$ containing $\mathbf{d}(t)$. From equation 2, $N \subset \mathcal{R}_{new}$ and thus $\mathbf{d}(t) \notin \delta\mathcal{R}_{new}$ by the definition of a boundary point.

Now consider the points on $\mathbf{c}$ that are deleted. Because $\mathbf{c} \in \delta M$ in a neighborhood of $x$, $\mathbf{d} \subset \delta\mathcal{R}$ in a neighborhood of $x$. It follows that $\big(\forall s \in (s_1, s_2')\big)\big(\mathbf{c}(s) \in \mathcal{R}\big)$ since $\mathcal{R}$ lies on the left of $\mathbf{d}$ and $M$ lies on the left of $\mathbf{c}$ (Figure 5(b)). Consider a point $\mathbf{c}(s)$, where $s \in (s_1, s_2')$. Because the motion regions are disjoint and the boundary curves do not touch, there exists a neighborhood $N$ of $\mathbf{c}(s)$ with $N \subset \mathcal{R} \cup M$ and $N \cap M' = \emptyset$ for all other motion regions $M'$. Then, from equation 2, $N \subset \mathcal{R}_{new}$ and thus $\mathbf{c}(s) \notin \delta\mathcal{R}_{new}$.

Each of the other cases can be proved similarly. □

**Directive 2.** Any closed curve $\mathbf{c}$ completely contained in $\overline{M}$, the closure of the motion region, should be deleted.

**Lemma 2.** *No point deleted by Directive 2 is in* $\delta\mathcal{R}_{new}$.

The proof is straightforward and is omitted.

**Theorem 3.** *Let* $D_1$ *and* $D_2$ *be the sets of curve points deleted by Directives 1 and 2 respectively. Then,*

$$\delta'\mathcal{R} - (D_1 \cup D_2) = \delta\mathcal{R}_{new}.$$

*Proof.* Let $D = D_1 \cup D_2$. From the definition of the motion regions and of $\mathcal{R}_{new}$, it is clear that $\delta\mathcal{R}_{new} \subseteq \delta'\mathcal{R}$. From the lemmas above, we have that $\delta\mathcal{R}_{new} \cap D = \emptyset$, so that $\delta\mathcal{R}_{new} \subseteq \delta'\mathcal{R} - D$.

Let $x \in \delta'\mathcal{R} - D$. Now either $x \in \delta\mathcal{R}_c$ or $x \in \mathbf{c}'_m$, for some $\mathbf{c}'_m$ bordering a motion region. Suppose $x \in \delta\mathcal{R}_c$. Then $x$ is a limit point of $\mathcal{R}$ and is not in $\mathring{\mathcal{R}}$. Also, $x \notin D$ means that $x$ is not in the interior of a motion region. It follows that $x$ is a limit point of $\mathcal{R}_{new}$ and is not in the interior of $\mathcal{R}_{new}$, which means $x \in \delta\mathcal{R}_{new}$.

Suppose $x \in \mathbf{c}'_m$, where $\mathbf{c}'_m \in \delta'\mathcal{R}$ borders an additive motion region $M$. Now, $x \notin D$ means that $x \notin \mathring{\mathcal{R}}$. Also, $x \in \delta M$ means that $x$ is not in the interior of any motion region. Thus, $x$ is not in the interior of $\mathcal{R}_{new}$, but $x$ is a limit point of $M$ and thus a limit point of $\mathcal{R}_{new}$. Therefore, $x \in \delta\mathcal{R}_{new}$.

Similar reasoning holds when $x$ is on the boundary of a subtractive motion region. ☐

### 2.4 Determining the topology

The ideas presented above can be used to ensure that the boundary contours correctly represent the region even when the topology of the region changes. They do not, however, tell us what the new topology is. The topology of an arbitrary region cannot in general be determined with purely local information [11]. We can nevertheless use temporal knowledge to avoid examining all the points in the boundary representation. Consider an additive motion region through which a single curve portion passes (e.g. Figure 2(e)). If the two intersecting curve portions $c_1$ and $c_2$ belong to the same boundary curve then, because the motion region is an additive region, the (sub)region has wrapped around on itself and has thus created an extra hole. On the other hand, if $c_1$ and $c_2$ belong to two different boundary curves, then two subregions have merged, and thus a hole disappeared or two previously disjoint regions are now joined. The problem here is that one cannot maintain the necessary information in a local manner. For example, suppose each curve portion had a tag indicating the boundary curve to which it belongs. If a single curve splits, then all the portions on one of the new curve must have their tag updated; this is clearly not local. It is not completely global, however, as only the curve involved in the split need be modified. This is what we call pseudo-local.

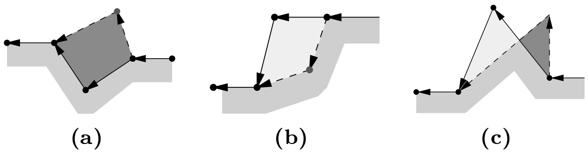



**Figure 6:** The medium grey area shows the points that are part of the original region. The dark grey indicates an additive motion region, while the light grey indicates an subtractive motion region.

## 3 Polygonal implementation

We have implemented the ideas above using a polygonal representation for the boundary. This section briefly describes the implementation. The boundary contours are represented by a set of polygons which are in turn represented by a set of vertices. A "motion function" is used to determine a motion vector for each vertex. The results presented in Section 4 use the DBM-Estimator objective function [14].

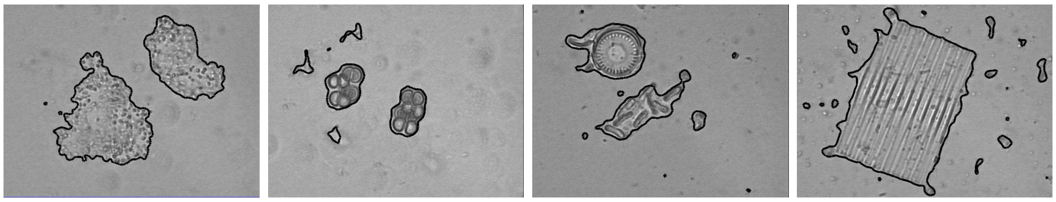
The algorithm alternately moves a vertex by applying a motion vector and corrects the representation if necessary. Since only a single vertex is moved, the motion region is always a quadrilateral (Figure 6). In most cases the quadrilateral is simple and there is a single motion region. When the quadrilateral is not simple, we consider it to form two disjoint motion regions, one additive and one subtractive.

To correct the representation, we first compute all intersections caused by the movement. The search for intersecting edges can be localized to area around the moving vertex by constraining the maximum edge length. The edges in this area can be efficiently retrieved by storing the vertices in bins indexed by their location. There are typically very few edges in the area (less than 5), and so the intersections can be computed in $O(1)$ time in the common case.

The next step is to apply the directives presented above. Directive 1 is implemented efficiently using a few dot and cross products because the intersections arise from line segments and because the motion region is a quadrilateral. Directive 2 is implemented efficiently using the vertex bins to find contours completely contained in the motion region. Both directives take $O(m+1)$ time in the common case, where $m$ is the number of edges that need to be deleted.

## 4 Results

By simply combining the algorithm above with an energy function to direct the boundary's motion, one obtains an active contour that maintains valid boundary representations and properly handles topological changes. Here we present results using energy func-



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

**Figure 7:** A single contour initialized ten pixels away from the image boundary shrinks and splits to tightly wrap the phytoplankton.

tions defined by a DBME objective function [14] as applied to two different applications: surface extraction and phytoplankton classification.

Given a model and a seed point, the surface extraction problem is to (a) extract the maximal connected region matching the model and (b) to determine the model parameters. Here we extract planar surfaces from range images. Figure 1 shows how the algorithm proceeds given a seed point on a ring. In our experiments, the energy function often generates motions that cause a single intersection (Figure 2(f)) and sometimes even complete reversals (Figure 2(g)). This occurs more often when a contour smoothing term is not used. Having an algorithm capable of handling these changes in topology and maintaining a valid boundary representation is essential in extracting (and correctly describing) complex surfaces.

The second application is phytoplankton classification. The ability to robustly extract multiple organisms from noisy images is important for this task. For this application, we initialize a single rectangular polygon a few pixels away from the image boundary and "shrink-wrap" the phytoplankton. As the active contour evolves and splits, the colonies form regions of their own and are thus isolated from the background and each other (Figure 7). The ability to change topology is crucial in detecting multiple organisms in a single image because it allows the contours the flexibility to split and wrap tightly around the objects without any prior knowledge of the locations and shapes of the organisms. The tight wrap also enables us to easily discard the clutter (such as the small regions in Figures 7(c) and 7(d)) and concentrate on classifying the phytoplankton.

## References


[1] Blake, A., Curwen, R., and Zisserman, A. "A framework for spatiotemporal control in the tracking of visual contours." *Int. J. of Computer Vision* **11**(2) pp. 127–145, 1993.

[2] Cohen, L. D. and Cohen, I. "Finite-element methods for active contour models and balloons for 2-D and 3-D images." *IEEE Trans. on PAMI* **15**(11) pp. 1131–1147, November 1993.

[3] Ivins, J. and Porrill, J. "Active region models for segmenting textures and colours." *Image and Vision Computing* **13**(5) pp. 431–438, 1995.

[4] Ivins, J. and Porrill, J. "Constrained active region models for fast tracking in color image sequences." *Computer Vision and Image Understanding* **72**(1) pp. 54–71, October 1998.

[5] Kass, M., Witkin, A. P., and Terzopoulos, D. "Snakes: Active contour models." *Int. J. of Computer Vision* **1**(4) pp. 321–331, January 1988.

[6] Lobregt, S. and Viergever, M. A. "A discrete dynamic contour model." *IEEE Transactions on Medical Imaging* **14**(1) pp. 13–23, 1995.

[7] Malladi, R., Sethian, J. A., and Vemuri, B. C. "Shape modeling with front propagation: A level set approach." *IEEE Trans. on PAMI* **17**(2) pp. 158–175, February 1995.

[8] Markowsky, G. and Wesley, M. A. "Fleshing out wire frames." *IBM J. Res. Develop.* **24**(5) pp. 582–597, September 80.

[9] McInerney, T. and Terzopoulos, D. "T-Snakes: Topology adaptive snakes." *Medical Image Analysis* To be published.

[10] McInerney, T. and Terzopoulos, D. "Topologically adaptable snakes." In *Proceedings ICCV '95*, pp. 840–845. Cambridge, MA, June 1995.

[11] Minsky, M. L. and Papert, S. A. *Perceptrons: Introduction to Computational Geometry*. MIT Press, expanded edition edition, 1988.

[12] Munkres, J. R. *Topology: a first course*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

[13] Osher, S. and Sethian, J. "Fronts propagating with curvature dependent speed: algorithms based on the Hamilton-Jacobi formulation." *J. Comput. Phys.* **79** pp. 12–49, 1988.

[14] Paragios, N. and Deriche, R. *Geodesic Active Regions for Texture Segmentation*. Rapport de recherche 3440, INRIA, June 1998.

[15] Stewart, C. V., Bubna, K., and Perera, A. "Estimating model parameters and boundaries by minimizing a joint, robust objective function." In *Proceedings CVPR '99*. Fort Collins, CO, June 23–25 1999.