Dual and Topologically Adaptable Snakes and Initialization of Deformable Models

GILSON A. GIRALDI[1], N. VASCONCELOS[2], EDILBERTO STRAUSS[2], ANTONIO A. F. OLIVEIRA[1]

[1]National Laboratory of Scientific Computation
Av. Getulio Vargas, 333, 25651-070 Rio de Janeiro, RJ, Brazil
{gilson}@lncc.br [2]Federal University of Rio de Janeiro
System Engineering and Comp. Science Program - COPPE
Mail Box 68511, CEP 21945-970, Rio de Janeiro, RJ, Brasil
{strauss,oliveira}@lcg.ufrj.br, {nivaldo}@cos.ufrj.br

Abstract. The original proposal of Active Contour Models, also called snakes, suffers from the strong sensitivity to the initial contour position and can not deal with topological changes. In this paper we describe some techniques to address these limitations. The problem of topological changes is addressed by the T-Snakes model by embedding the snake in the framework of a simplicial decomposition of the domain. The sensitivity to initialization is addressed by the Dual method by using two contours going towards the desired boundary. We integrated these two methods, given a new approach called Dual-T-Snakes, which maintains the topological capabilities of the T-Snakes and the ability of avoiding local minima of the Dual Active Contour model. One advantage of the Dual-T-Snakes is the reduction of the search space, which allows a more efficient application of a dynamic programing technique. To initialize the Dual-T-Snakes automatically we developed a methodology based on a multiresolution scheme and on the T-Snakes framework. Just as the T-Snakes, our methods (Dual-T-Snakes and initialization framework) can be extended to 3D. We demonstrate our methodologies in synthetic and medical images showing also how to incorporate fuzzy and multi-scale techniques in the Dual-T-Snakes approach. Finally, we discuss some challenges in active contour models and present the conclusions of our work.

1 Introduction

Active Contour Models (*Snakes*), were proposed by Kass at al. [23] and since then have been successfully applied in a variety of problems in computer vision and image analysis, such as edge and subjective contours detection, motion tracking and segmentation [4].

Basically, there are two types of snake models: the implicit ones and the parametric ones.

Implicit models, such as the formulation used in [30], embeds the snake as the zero level set of a higher dimensional function and solves the corresponding equation of motion. Such methodologies are best suited for the recovery of objects with complex shapes and unknown topologies. However, due to the higher dimensional formulation, implicit models are not as convenient as the parametric ones, for shape analysis and visualization, and for user interaction.

The parametric snake models can be described such as an elastic curve (or surface) which can dynamically conform to object shapes in response to both internal forces (elastic forces) and external forces (image and constraint forces). These forces can be the result of a functional global minimization process or based on local information. Such approach is more intuitive than the implicit models. Its mathematical formulation makes easier to integrate image data, an initial estimated, desired contour properties and knowledge-based constraints, in a single extraction process [4].

However, parametric models have also their limitations. First, most of these methods can only handle topologically simple objects. The topology of the structures of interest must be known in advance since the mathematical model can not deal with topological changes without adding extra machinery[31]-[9]. Second, parametric snakes are too sensitive to their initial conditions due to the nonconvexity of the energy functional and the contraction force which arises from the internal energy term [44, 18].

Several works have been done to address the mentioned limitations.

The use of simulated annealing for minimization was proposed in [39]. Despite the global optimization properties, the use of this technique is limited by both its computational complexity and memory requirements.

Levine at al. [26] applied hierarchical filtering methods, as well as a continuation method based on a discrete scale-space representation. At first, a scale-space scheme is used at a coarse scale to get closer to the global energy minimum represented by the desired contour. In further steps, the optimal valley or contour is sought at increasingly finer scales.

These methods address the nonconvexity problem but not the bad effects of the internal normal force. This force is a contraction force which makes the curve collapse into a point if the external field is not strong enough.

In Cohen [7] and Gang Xu at al. [44] this problem is addressed by the addition of another internal force term to reduce the bad effects of the contraction force. In both works the number of parameters are increased if compared with the original model and there are some trade-offs between efficiency and performance.

Another way to remove the undesired contraction force of the original snake model is to use the concept of invariance which is well known in the field of computer vision [16]. This concept has been applied for closed contours, and consists in designing an internal smoothing energy, biased to ward some prior shape, which has the property of being invariant to scale, rotation and translation. In these models, the snake has no preference to expand or contract, but it tends to acquire a natural shape.

An example of a technique, which applies invariance concepts, is the Dual Active Contour (*Dual ACM)* [18]. This approach basically consists of one contour which expands from inside the target feature, and another one which contracts from the outside. The two contours are interlinked to provide a driving force to carry the contours out of local minima, which makes the solution less sensitive to the initial position.

These works do not address the topological limitations of the traditional snake model. We are interested in two topological operations: splits and merges. The proposals found in the literature to accomplish this are basically composed by two steps: first a procedure to identify the necessity of a topological operation and second a (surgical) procedure to implement it [31]-[9].

The T-Snakes model is an example of such a methodology which has the advantage of being a general one. The method embeds the snake model within a framework defined by a simplicial domain decomposition, using classical results in the field of combinatorial topology [1]. The resulting model has the power of an implicit formulation without the need of a higher dimensional formulation.

In the original proposal of the T-Snakes, the non-convexity of the energy is addressed by using region-based statistics to weigh a force used to make the snake converge to the boundaries.

In [12] we propose to address this problem by using a dual active contour framework. We demonstrate how to incorporate to the T-Snakes model a *generalized dual framework*: one T-Snake contracts and splits from outside the target(s) and the other one(s) expand(s) from inside during the process of seeking for the objects boundaries in a scene.

As in [18] our dual snakes uses the idea of comparing contours energy and positions in the decision of whether an equilibrium position is the final solution. However, to allow the utilization of more than one inner curve we use a normalization of the external (image) energy and an $affinity$ restriction. Also we takes advantage of the simplicial domain decomposition framework of the T-Snakes to compare curves instead of using Fourier descriptors as in [18].

This new approach - Dual-T-Snakes - maintains the topological capabilities of the T-Snakes and the ability of avoiding local minima of the Dual ACM. Our proposal can be viewed as an alternative to the statistical approach originally proposed in the T-Snakes [34], or as a complementary one.

The sensitivity to initialization of snakes can be addressed also by a two stage approach: (1) the region of interest is limited; (2) a global minimization technique is used to find the object boundary. Few works have been done in this direction. In [3] is described such a method to segment cell nucleus based on a dynamic programming algorithm (*Viterbi Algorithm*) to find the solution.

The use of *Dynamic Programming* (*DP*) for solving variational problems is discussed by Amini at al. [2]. Unlike the variational approach, DP ensures global optimality of the solution and do not requires estimates of higher order derivatives which improves the numerical stability. However, these techniques are limited by their storage requirements of $O\left(NM^2\right)$ and computational complexity of $O\left(NM^3\right)$, where N is the number of snaxels and M the size of the neighborhood around each snaxel (given a $discrete$ $search\ space$ with NM points). These performance difficulties can be lowered with a method to reduce the search space. That is the main point addressed in [14].

In that work, we propose to reduce the search space through the Dual-T-Snakes model [12] by its ability to get closer to the boundaries desired. The result is two contours close to the border bounding the search space. Hence, a DP algorithm [2, 3, 19] can be used more efficiently.

The sensitivity to the initial position can also be addressed by a method which initializes automatically the snake closer to the boundaries.

The automatic initialization of snake models is a problem with few references in the literature [25]. However, an efficient methodology in this field would be worthwhile not only to save time calculation but also to facilitate the specification of parameters.

In [15, 11] we propose a method to initialize deformable models which is based on properties related to the topology and spatial scale of the objects in 2D or 3D scenes. We assume some topological and scale properties for the objects of interest. From these constraints we propose a method which first defines a triangulation of the image domain. After that, we take a subsampling of the image field over the triangulation grid nodes. This field is thresholded, generating a binary one, an "Object Characteristic Function", from which a rough approximation of the boundary geometry is extracted.

We demonstrate our methods (Dual-T-Snakes and initialization framework) in 2D synthetic and medical images showing also how to incorporate fuzzy and multi-scale techniques in the Dual-T-Snakes approach.

Despite of the intense research to address the limitations of the original snake model [23] as well as extend its range of applications, there are still some open problems and challenges in the field. The utilization of neural networks for initializing snake models is the first one we discuss. Next, we present recent works on integration of snake models and the *live wire* - a user steered boundary extraction paradigm.

After that, we present the open problem of tracking objects which undergo topological changes during their evolution. Finally, we discuss the integration of a Snakes System in a *Modular Visualization Environments*, like Khoros [6], to take advantage of the software integration and development of these systems as well as to give them another tool (*toolbox*) for boundary extraction and segmentation.

This paper is organized in the following way. In section 2 we discuss the original snake model. In sections 3 and 4 we present the basic models for our work. From sections 6 to 8 we summarize our contributions: the Dual-T-Snakes model and an initialization framework for deformable models. We show 2D results in section 9. The Appendix A gives a summary of the the multiseed fuzzy connectedness segmentation which can be integrated with our work in a natural way. Finally, we present future works and conclusions.

2 Classical Snake Model

The snake model [23] is a kind of deformable model which and can be used to process 2D or 3D data [4].

According to Fisker at al. [37], deformable models can roughly be classified into two groups: *Free Form and Parametric*.

In parametric deformable models prior knowledge of the global structure is included using a parameterized template of a specific structure. Free Form deformable templates, like snakes, have no explicit global structures as the prior knowledge contains basically local continuity and smoothness constraints.

Considering as a functional energy minimization process then snake models consists of an initial model which is carried to the desired object boundary by forces described by the Euler-Lagrange equations. In a different way, the snake evolution can be formulated by local deformations to reshape dynamically the initial model in a process which do not apply minimization techniques explicitly.

The former formulation is the one used by Kass at al. [23] as we describe next.

2.1 Original Model

Geometrically, a snake is a parametric contour c, here assumed to be closed, embedded in a domain $D \subset \Re^2$:

$$c : [0,1] \to D \subset \Re^2; c(s) = (x(s), y(s)). \tag{1}$$

We can define a deformable model as a space of admissible deformations (contours) Ad and a functional E to be minimized [7]. This functional represents the energy of the model and has the form:

$$\begin{aligned} E &: Ad \to \Re; \\ E(c) &= E_1(c(s)) + E_2(c(s)); \end{aligned} \tag{2}$$

where

$$E_1 = \int_\Omega \left(w_1 \left\| c'(s) \right\|^2 + w_2 \left\| c''(s) \right\|^2 \right) ds\,, \tag{3}$$

$$E_2 = \int_\Omega P(c(s))\, ds, \tag{4}$$

are the internal and external energy terms, respectively. In the internal energy, the parameter w_1 (tension) gives the snake the behavior of resisting to the stretch and w_2 (rigidity) makes the snake less flexible and smoother. These parameters can be constants or dependent on s [26]. Each prime denotes a degree of differentiation with respect to the parameter s.

In the external energy E_2, P is the potential associated with the external (image) forces and is narrow related with the features we seek. For edge detection in a greyscale image a possible definition is [4]:

$$P = -\left\| \nabla I \right\|^2, \tag{5}$$

where I is the image intensity.

The process of minimizing the functional given in (2) can be viewed from a dynamic point of view by using the Lagrangian mechanics . This leads to dynamic deformable models that unify the description of shape and motion. In these models the deformable contour is viewed as a time-varying curve

$$c(s,t) = (x(s,t), y(s,t)), \tag{6}$$

with a mass density μ and a damping density γ.

In this formulation, the Lagrange equations of motion for a snake with potential energy given by expression (2) have the form [26, 34]:

$$\mu \frac{\partial^2 c}{\partial t^2} + \gamma \frac{\partial c}{\partial t} + \left(w_1 c'(s)\right)' + \left(w_2 c''(s)\right)'' + \nabla P(c(s)) = 0, \tag{7}$$

where the first two terms represent the inertial and damping forces while the third and fourth terms give the forces related to the internal energy (equation (2)). The last term in equation (7) is the external force due the external potential P in expression (5). The equilibrium is achieved when the internal and external forces balance and the contour comes to rest; which implies that:

$$\partial c/\partial t = \partial^2 c/\partial t^2 = 0. \quad (8)$$

In order to solve the equation (7) for an initial closed contour we have to discretize the snake in space and time by using local (*Finite Differences*) or global representation (*Finite Elements*) methods each of them with trade-offs between performance and numerical efficiency [26, 8]. We have also to use a *termination condition* to stop the numerical interactions [26].

It is important to observe that the space Ad in expression (2) do not include contours with more than one connected component. So the classical snake model do not incorporate topological changes of the contour c during its evolution given by equation (7). Besides, the contraction force generated by the third and fourth terms in this equation is shape dependent and makes the stabilization of the snake too dependent of the parameters w_1 and w_2. While in theory it is possible to compute a pair of proper weights of the internal energy for each point, it is very difficult in practice [44]. The following technique shows a way of addressing this problem by using an invariant internal energy.

3 Dual Active Contour Model

In this model [18] and for the discussion that follows a snake will be considered a particle system $v_i = (x_i, y_i)\,, i = 0, ..., N-1$ whose particles are linked by internal constraints to form a closed polygonal. In fact, such formulation can be viewed as a discrete form of the classical formulation described above.

The basic idea of the Dual Active Contour Model (*Dual ACM*) is to reject local minima by using two contours: one which contracts from outside the target and one which expands from inside [18]. Such proposal makes possible to reduce the sensitivity to initialization, by enabling a comparison between the two contours energy, which is used to reject local minima. To obtain the conventional continuity and smoothness constraints, but removes the unwanted contraction force, a scale invariant internal energy function is developed.

Specifically, the shape model is accomplished by the following internal energy [18]:

$$E_{int} = \frac{1}{2}\sum_{i=0}^{N-1}\left(\frac{\|e_i\|}{h}\right)^2, \quad (9)$$

where:

$$e_i = \frac{1}{2}\left(v_{i-1} + v_{i+1}\right) - v_i + \frac{1}{2}\theta_i R\left(v_{i-1} - v_{i+1}\right), \quad (10)$$

R is a 90^0 rotation matrix and θ_i is related to the internal angle φ_i in the vertex v_i by:

$$\theta_i = \cot\left(\frac{\varphi_i}{2}\right). \quad (11)$$

It is clear that E_{int} has a global minimum when $e_i = 0, i = 0, 1, ..., N-1$. From (10)-(11) it can be shown that this happens when:

$$\varphi_i = \pi\left(N-2\right)/2N, i = 0, 1, ..., N-1, \quad (12)$$

which are the internal angles of a regular polygon with vertices given by the points v_i [18]. The energy (9) can be also shown to be rotation, translation and scale invariant [18].

The total energy of the model is given by:

$$E = \frac{\lambda}{N}E_{int}\left(v_i\right) + \left(1-\lambda\right)E_{ext}\left(v_i\right), \quad (13)$$

where E_{ext} is the discrete form of (4) and λ is a smoothing parameter which lies between 0 and 1 [18].

With the internal energy (9) the internal contraction force problems are avoided because the curve do not have the tendency to collapse but only to satisfy the constraint (12) - the curve is biased to ward a regular polygon. Also, this fact makes easier to establish the correspondence between the points of the two contours because the form of the snake during the evolution is limited by the energy (9). The methodology takes advantage of this correspondence by proposing the *driving force*:

$$F_{drivng} = g\left(t\right)\frac{u_i - v_i^t}{\|u_i - v_i^t\|}, \quad (14)$$

where v_i^t is the contour being processed at time t, u_i is the contour remaining at rest and $g\left(t\right)$ is the strength of the force.

The termination condition adopted in [18] is a discrete version of equation (8) when the model has no inertia (mass $\mu = 0$):

$$\max_{i}\ \left\|v_i^{t+1} - v_i^t\right\| < \delta, \quad (15)$$

where δ is a termination parameter.

The dual approach consists in making the inner and outer contours evolve according the following algorithm: The contour with the highest energy (13) is selected. If its motion remains below some termination condition then the driving force (14) is increased until it moves at a rate greater than the chosen threshold δ . When the energy begins to decrease, the added driving force is removed and the contour is allowed to come into equilibrium. The procedure is then repeated until both contours have found the same equilibrium position.

4 Topologically Adaptable Snakes

By topological changes we mean the ability of a contour to split into two or more snakes or N contours to merge into another one.

In the field of implicit snake models such operation are inherent due the higher dimensional mathematical formulation they use [30].

For parametric snake models extra machinery has to be added to the model. First of all, it is easier to formulate the snake evolution by local deformations because the definition of a global energy in this case is very difficult.

In [40] we found such a methodology which seeds particles on the surface of an object until their density on the surface is within some threshold value. The model uses inter-particle forces to maintain the smoothness of the surface (or contour) and external image forces to attract particles to the image features. This technique has the advantage of dealing easily with open and closed surfaces. However it has the disadvantages of being expensive ($O\left(N^2\right)$ where N is the number of particles) and that may be difficult or cumbersome to find good initial seed particle sites automatically [34].

In [9] we found a methodology to split a closed contour in two closed parts. This is accomplished by first constructing a histogram of the image force intensity along the snake to identify the appropriate region to cut it (region with weakest image field). Next, the method identifies two points in this region to be the end points of the segment which will cut the curve into two parts. The criterium to do this is based on the direction of an area force used to make the contour fit concave parts. This methodology has the disadvantages of not dealing with the contour merges and its extension to 3D is very difficult.

A more general approach to incorporate topological changes in the parametric snake models is the T-Snakes framework [31]-[34]. Next, we review the basic elements of this method which is fundamental for our work.

4.1 T-Snakes: A framework for topological Changes

The T-Snakes approach is composed basically by three components: a simplicial decomposition of the domain of interest, in our case a closed subset $D \subset \Re^2$, a particle model of the snake and a *characteristic function* defined over the simplicial which distinguishes the interior from the exterior of a contour c:

$$\chi : D \subset \Re^2 \rightarrow \{0, 1\} \tag{16}$$

such that: $\chi\left(p\right) = 1$ if $p \in Int\left(c\right)$ and $\chi\left(p\right) = 0$, otherwise, where p is a node of the simplicial grid.

Starting from these elements, topological operations can be reduce to find a piecewise linear representation of the final contour as we shall see bellow. Such proposal has the advantage of importing classical results of combinatorial topology [1, 34] to the context of active contours models given a general and simple methodology to deal with topological operations.

4.1.1 Piecewise Linear Representations

The concepts to be presented are basic to get piecewise linear representations of differential manifolds [1] implicitly defined accordingly to the known *Implicit Function Theorem*. This theorem states that for a function $H : \Re^{N+K} \rightarrow \Re^N$, the set:

$$M = \left\{x \in \Re^{N+K} \mid H\left(x\right) = 0, \text{x is a regular point of} H\right\}, \tag{17}$$

is a differential manifold of dimension K.

Following [1], the first point to get a piecewise linear representation for the set M above is to take a domain decomposition. Given a closed subset $D \subset \Re^n$ there are two main types of domain decomposition methods: non-simplicial and simplicial.

The non-simplicial methods are based on a cell decomposition of the space (Marching Cubes [29] is an example) and so they can not be used to represent surfaces or contours unambiguously without a priory well defined disambiguation scheme.

Simplicial decompositions, also called (*Triangulations*), on the other hand, provide an unambiguous framework for the creation of local polygonal approximation of a contour or surface [1].

The next step is to define a piecewise linear approximation of the function H over a simplicial decomposition (triangulation for simplicity) of $H : \Re^{N+K} \rightarrow \Re^N$. For our interest, the function H is given by a Characteristic Function like in expression (16).

Next, we present some definitions well known in the subject of combinatorial topology [1] which are basic for this work.

4.1.2 Piecewise Linear Methods

Let's suppose a triangulation of an Euclidean space E of finite dimension. Following the classical nomenclature, a vertex of a triangle (simplex) will be called a *node* and the collection of nodes and edges will be called the *simplicial grid* Γ_s.

Definition 1. $tng(\sigma) = aff(\sigma) - \sigma$ denotes the tangent space of σ which is the vector space obtained by translating the *affine hull* $aff(\sigma)$ [1] to the origin.

Definition 2. If $u \in E$, $u \neq 0$ and $\gamma \in \Re$, then the set $\{x \in E \mid u^{*}x \geq \gamma\}$ is called *half-space* and its boundary $\{x \in E \mid u^{*}x = \gamma\}$ a hyperplane. A finite intersection of half-spaces is called a convex *polyhedral* set or a *cell*. Hence, cells are closed convex sets which may be bounded or unbounded. A bounded cell is called *polytope*.

Definition 3. The dimension of a cell is identified with the dimension of its tangent space, that is, $\dim(\sigma) = \dim(tng(\sigma))$. If this dimension is m, we say that the cell is *m-dimensional* or simply that we have a *m-cell*.

Definition 4. A $face$ τ of a cell σ is a convex subset $\tau \subset \sigma$ such that for all $x, y \in \sigma$ and $\lambda \in \Re$ the following property holds: $x, y \in \sigma$, $0 < \lambda < 1$, $(1-\lambda)x + \lambda y \in \tau \Longrightarrow x, y \in \tau$.

Trivially, a cell σ is a face of itself. All other faces of σ are called *proper faces*. A proper face τ of maximal dimension, that is $\dim(\tau) = \dim(\sigma) - 1$, is called a facet of σ.

A face of dimension k is called a *k-face*.

Definition 5. Piecewise Linear Manifolds: Let $\widehat{M}$ a family of $(N+1) - cells$ For $0 \leq k \leq N+1$, we define the sets:

$$M = \left\{x \in \Re^{N+K} \mid H(x) = 0, \text{x is a regular point of} H\right\}, \quad (18)$$

$$\widehat{M}^{k} = \left\{\tau \mid \tau \text{ is a k-face of some cell} \sigma \in \widehat{M}\right\}$$

and:

$$\left|\widehat{M}\right| = \bigcup_{\sigma \in \widehat{M}} \sigma.$$

We call $\widehat{M}$ a *Piecewise Linear Manifold* of dimension $N+1$ if and only if the following conditions hold:

(1) The intersection $\sigma_1 \cap \sigma_2$ of two cells $\sigma_1, \sigma_2 \in \widehat{M}$ is empty or a common face of both cells;

(2) A facet $\tau \in \widehat{M}^{N}$ is common to at most two cells of $\widehat{M}$;

(3) The family $\widehat{M}$ is locally finite, that is, any compact subset of $\left|\widehat{M}\right|$ meets only finitely many cells of $\widehat{M}$.

The fundamental goal in this theory is to represent a manifold defined by the *Implicitly Function Theorem* through a PL Manifold with some degree of precision [1]. For our interest, the PL manifolds will be defined through a Characteristic Function like the equation (16). For such a function, the following definitions will be useful.

Definition 6. We call an $N - face$ τ of a triangle σ *completely labeled* respected to χ if this function changes its value in τ.

Definition 7. A triangle σ in $\Re^{N+K}$ is called traverse with respect to χ if it contains a completely labeled $N - face$.

Finally, we can state a fundamental theorem for our work.

Theorem 1. If $\sigma \in \Gamma$ has a non-empty intersection with $\chi^{-1}(0)$, then $M_\sigma = \sigma \cap \chi^{-1}(0)$ is a um $K - dimensional$ polytope, and the family

$$M_\Gamma = \left\{M_\sigma \mid \sigma \in \Gamma, \sigma \cap \chi^{-1}(0) \neq \emptyset\right\},$$

is a Piecewise Linear Manifold of dimension K (*K-dimensional PL Manifold*).

Dem. See [1].

If $\chi : \Re^{1+2} \rightarrow \Re$ then the possible polytopes are the ones on Figure 1:

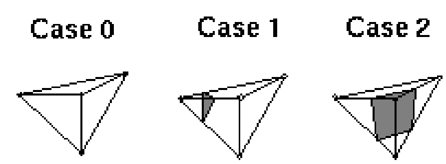


Figure 1: Reduced polytopes in 3D.

The first point to observe is that the image domain is a compact subset of $\Re^3$. So, item (3) in the PL Manifold definition above guarantee that the number of polytopes is finite. Also, the theorem above guarantees that the surface generated has no holes. So, the solutions are closed surfaces interior to the image domain or open surfaces intersecting the image frontiers.

If we use CF triangulations like in Figure 2, we can take advantage of the simple rules to go from one simplex to an adjacent one (*pivoting steps*) and so using optimized algorithms to generate the corresponding PL Manifolds ([1], pages 210 and 240).

For $\chi : \Re^{1+1} \rightarrow \{0, 1\}$, the following theorem simplifies the PL Manifold generation:

Property 1 (*Door-In-Door-Out Principle*). A triangle has either zero or exactly two completely labeled *1-faces*.

Based on this property we can generate a 1-dimensional PL Manifolds by first finding a starting traverse triangle and then use one of its completely labeled edge as the *door-out* to go towards the next traverse triangle. The algorithm can be summarized as follows:

Continuation Algorithm (Door-In-Door-Out)
input
begin
. $\sigma_0 \in \Gamma$ traverse triangle (initial simplex)
. τ_0 n-face completely labeled of σ_0;
end;
Repeat:

. find $\sigma_n \in \Gamma$, $\sigma_n \neq \sigma_{n-1}$ such that $\tau_{n-1} = \sigma_n \cap \sigma_{n-1}$;

. find a n-face completely labeled τ_n de σ_n such that $\tau_n \neq \tau_{n-1}$;

. until returning to initial simplex $\sigma_0 \in \Gamma$.

The sequences of triangles $\sigma_0, \sigma_1, ..., \sigma_q$ and edges τ_0,τ_1, ..., τ_q so generated, have pairwise different elements [1]. Also if D is compact, we can prove the following property [1]:

Property 2. The sequences $\{\sigma_i\}$, $\{\tau_i\}$ are finite or there is a finite integer $q > 0$ such that $\sigma_0 = \sigma_q$.

4.1.3 Snakes Reparameterization

In this framework, the reparameterization of a contour is done by taking the points of the contour as the intersections points of the snake with the simplicial grid and the topological changes are carried out by distinguishing the interior and exterior of the snake(s). The Figure 2 shows the simplest triangulation of the plane - the Coxeter-Freudenthal triangulation- which is constructed by partitioning space using an uniform square grid and then subdividing each square into two simplices. The figure shows also a contour projected over the simplicial grid Γ_s corresponding.

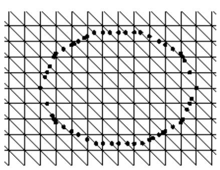


Figure 2: CF Triangulation and contour projected.

Let's see how these ideas were incorporated in the snake model to allow topological changes. Let us take the characteristic function (16) defined over a Coxeter-Freudental triangulation of the plane and two closed contours according the figure 3. The marked vertices have value 1 and the non-marked have the value zero according (16). If we use the door-in-door-out algorithm to trace the triangles with at least one (but not all) vertices outside the region surrounded by the contour(s) we can find a one dimensional PL manifold which represents the merge of the two curves. The same would be true for more than two contours (and obviously for only one). The properties 1 and 2 of section 4.1.2 show that this procedure is efficient and without ambiguities.

4.1.4 Snake Model

A T-Snakes is a discrete form of the classical snake described in section 2.1. It is defined as a set of N particles (snaxels) whose positions

$$\{v_i = (x_i, y_i), i = 0, ..., N-1\},$$

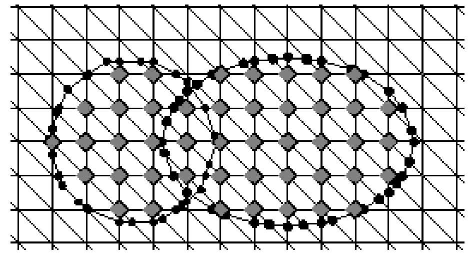


Figure 3: Two snakes colliding with the inside grid nodes and snaxels marked.

are connected in series to form a closed contour. The particles (snaxels) may have mass m_i and damping γ_i.

Each pair of points v_i, v_{i+1} is called a *model element*. These points are linked by a spring defined by a stiffness parameter a_i and a natural length l_i . The idea is the snake resists expansion or compression only when the actual length $\|r(t)\| = \|v_{i+1} - v_i\|$ is greater or less than l_i respectively. Hence, given the deformation $e_i = \|r_i(t)\| - l_i$, we define a tension force given by:

$$\alpha_i = a_i e_i r_i(t) - a_{i-1} e_{i-1} r_{i-1}(t). \tag{19}$$

In addition to the force (19) it is convenient to define a rigidity force to minimize the local curvature of the contour. In the T-Snakes model this force is defined by:

$$\beta_i = b_i \left(v_i - \frac{1}{2} (v_{i-1} + v_{i+1}) \right), \tag{20}$$

which attempts to minimize the distance between a point v_i and the centroid of its neighbors. The model has also a normal force

$$F_i = sig(v_i) \, k n_i, \tag{21}$$

where $sign(v_i) = -1$ if $|I(p) - \mu| > \sigma$ and $(v_i) = -1$ otherwise. In this definition, n_i means the normal at the snaxel v_i and k is a force scale factor and μ, σ are mean and variance for the image I. This force is used to push the model towards image edges until it is opposed by external image forces.

The forces given in (19)-(21) are internal forces. The external force is defined in function of the image data, according the features we seek. One possibility is the following definition:

$$f_i^t = -\nabla P, \tag{22}$$

where P is the potential definied on (5).

The evolution equation for the T-Snakes is finally given by:

$$v_i^{(t+\Delta t)} = v_i^t + \frac{\Delta t}{\gamma_i} \left(\alpha_i^t + \beta_i^t + F_i^t + f_i^t \right). \tag{23}$$

4.1.5 Classification Algorithm

During the T-Snake evolution some grid nodes become interior to a contour. Such nodes are called *burnt nodes*. To avoid the known problems that the evolution in the normal direction can bring (development of shocks and singularities [34]) the T-Snake model incorporates an *entropy condition*: *once a node is burnt it stays burnt* [34].

In a initial step, for each model element we consider the *box* made by the new and anterior positions (Figure 4).

Figure 4: Polygon with interior grid node (burned).

For each node in the box interior the image is divided in four subspaces through the semi-lines given in Figure 5.

Figure 5: Subspaces used in the classification algorithm.

During an evolution step, p_1 and p_2 are moved for new positions p_{1n} and p_{2n}, respectively. The step of a contour during a deformation step is restricted to be much less than the dimension of the image domain [34]. So, we can assume that p_{1n} and p_{2n} belong to some of the subspaces of Figure 5. Then, we have sixteen possibilities to consider, as we can see on Figure 6.(1)-6.(16).

Figure 6: Cases to be considered in the classification.

The classification of a grid node between *burnt* and *not burnt* is finally obtained by considering the following definitions:

A point p is sad to be interior of a policy Q if a ray cast from p intersects exactly one edge of Q or exactly three edges of Q.

A grid node v is labeled as *burnet* if it is inside polygon Q.

According to these definitions, from Figure 6, cases 1, 2 and 7 would label v burnt, cases 3-6 and 8-12 would label v not burnt. Thus, by simply classifying p_{1n} and p_{2n} into one of the four spaces of Figure 5, we can automatically label v for the majority of cases. The ambiguous cases 13-16 require additional tests (see [34, 33] for details).

4.1.6 T-Snakes Algorithm

The T-Snake model can be summarized as follows [34]:

For each time step:

1. Compute the external and internal forces and update the snaxels positions using equation (23),

2. Compute the intersection between the grid and model elements,

3. Update the characteristic function (16),

4. Using the characteristic function to determine the corresponding set of traverse triangles and compute all new model elements and snaxels

5. Determine the model elements still valid (it belongs to a traverse triangle) . Discard the other ones.

A termination condition is obtained by asing a *temperature* to each model element based on the number of deformations steps that a simplex has remained a boundary one. A T-snake is considered to have reached its equilibrium state when the temperature of all the snaxels fall below a pre-set *freezing point*. Once a T-snake has reached its equilibrium the simplicial grid can be discarded, if desired, and the snake runs as a classical discrete snake model according to equation (23).

5 T-Surfaces

One of the advantages of the T-Snakes is that its extension to 3D maintains the same methodology of the 2D case. The corresponding method - *T-Surfaces* - can be used for surface reconstruction of objects with unknown topology in 3D images [33]. The theoretical elements of this method is the piecewise linear manifolds concepts of section 4.1.2.

As for 2D case, the basic elements of T-Surfaces are: image domain triangulation, a characteristic function and a discrete model for snakes in 3D.

Over a topological point of view, a T-Surface is a piecewise linear manifold of dimension 2. In a dynamic sense, a T-Snake is a closed elastic triangular mesh. The mesh points are called *node elements* and the mesh triangles are called *triangular element*.

To each node element v_i it is associated elastic forces, a normal balloon-like force, and an image force. The corresponding evolution equation has the same form of expression (23).

The reparameterization process follows the same steps described in the 2D approach. However, the updating process of the characteristic function has to be changed to deal with the 3D case.

The characteristic function for T-Surfaces has the same role that in the T-Snakes: to distinguish the interior from the exterior of a surface allowing an efficient way of doing topological changes.

The update of the characteristic function is again the kernel of the method. The algorithm to accomplish this is analog to that one of the T-Surfaces. However, in this case we have to deal with the prism defined by each triangular element $\{p_1, p_2, p_3\}$ in the anterior position and its corresponding $\{p_{1n}, p_{2n}, p_{3n}\}$ in the new one.

During its evolution each triangular element may pass over zero, only one or several grid nodes. The polyhedron defined by linking the points $\{p_1, p_2, p_3\}$ with their corresponding ones $\{p_{1n}, p_{2n}, p_{3n}\}$ is divided in eight parts (Figure 8). To find if a grid node was burnt during the surface evolution we have to determine if this node is inside some of these parts.

In the first step of the algorithm we divide the space in eight subspaces through the three planes defined by linking he two nodes of each edge of the triangular element in the actual position with the grid node (Figure 7).

Next, we classify the three element nodes $\{p_{1n}, p_{2n}, p_{3n}\}$ according to the eight subspaces (Figure 8). The result obtained is used to find easily if a grid node was burnt.

Once the characteristic function is updated, the algorithm to generate the connected components of the surfaces is an extension of the door-in-door-out (section 4.1.2) to 3D. It can be summarized as follows:

Initialization
    begin
    Find a traverse simplex $\sigma \in \Gamma$;
    Mount the queue of traverse simplices already found: $\sum = \{\sigma\}$;
    Mount $V(\sigma)$ =set of nodes (vertices) of σ;
    end;
while $V(\sigma) \neq \emptyset$ for some $\sigma \in \sum$ **do**
    begin
    get $\sigma \in \sum$ such that $V(\sigma) \neq \emptyset$;
    given $v \in V(\sigma)$;
    obtain $\widetilde{\sigma}$ from σ by pivoting the vertex v into $\widetilde{v}$
    if $\widetilde{\sigma}$ is not a traverse one
        then drop v from $V(\sigma)$.
    else
        if $\widetilde{\sigma} \in \sum$ **then**
            drop v from $V(\sigma)$ and $\widetilde{v}$from $V(\widetilde{\sigma})$.
        else
            begin
            $\sum \doteq \sum \cup \{\widetilde{\sigma}\}$;
            $V(\widetilde{\sigma})$ =set of vertices of $\widetilde{\sigma}$;
            drop v from $V(\sigma)$ and $\widetilde{v}$from $V(\widetilde{\sigma})$.
        end{else};
end{while};

Other possibility would be to use the Tetra-Cubes **??** algorithm which uses the fact that the intersections between a plane and a tetrahedron can be reduced to a few cases (Figure 1).

The T-Surfaces algorithm has the same steps of the T-Snakes algorithm. The triangulation used in [32, 34] is a Coxeter-Freudenthal tetrahedral grid due to its simplicity, mainly for the pivoting steps.

Figure 7: Subespaces used in 3D classification algorithm.

Figure 8: (a)Triangular element has moved to new position and a node was burnt. (b) Polyhedron defined by the triangular element before and after defomation step. (c) Space partition.

6 Dual-T-Snakes Approach

The key idea behind this method is to explore the T-Snake framework to propose a generalized Dual ACM approach: one contour contracts and splits from outside the target(s) and other contour(s) expand from inside the target(s) (see Figure 9).

First, as the initialization is in general too far from the targets, it is not meaningful to use an explicit minimization process to evolve the snake. So, the snake model of section 4.1.4 is adopted.

Also, we need elements that make the evolution of the T-snakes outside and inside the targets interdependent to turn their evolution an unified and effective methodology. This aim is accomplished in this work by the image energy and an $affinity$ restriction.

We use two different definitions for the image energy: one for the outer contour ((E_{inner})) and another one for the set of inner contours enclosed by it ((E_{outer})):

$$E_{outer} = \sum_{i=0}^{N-1} \left(- \left\| \nabla I \left(v_i \right) \right\|^2 \right) / N, \qquad (24)$$

$$E_{inner} = \frac{1}{m} \left\{ \sum_{k=0}^{m} \left(\sum_{i=0}^{N_k - 1} \left(- \left\| \nabla I \left(v_i \right) \right\|^2 \right) / N_k \right) \right\}, \qquad (25)$$

where m is the number of inner curves and N, N_k are the number of snaxels of the outer snake and of inner snake k, respectively.

If $E_{outer} > E_{inner}$ the outer snake evolves. Otherwise we have to decide which inner curve to chose. This can be accomplished by a notion of $affinity$: Given a snaxel, we define the affinity between a neighborhood of it and the region limited by the snake as the likelihood that they belong to the same object.

That likelihood is thresholded to obtain an affinity function that assigns to the snaxel a 0-1 value. It is important to emphasize that the concept of affinity is used here not specifically to improve the accuracy of the final solution as in [22]. Our aim here is mainly to avoid that a T-Snake which has already conformed to a boundary be chosen to be processed again.

The next point is the similarity between two contours. The idea is to take advantage of the T-Snakes framework as a way of comparing the positions of the snakes: the difference between the Characteristic functions of the outer snake and set of inner ones ($Characteristic\_Diff$) gives a notion of (local) proximity between them. For example, in the case of the CF triangulation of the Figure 3 we can stop the motion of all snaxels of an inner contour inside a triangle if any of its vertex v has the two following properties: 1) All the 6 triangles adjacent to v have a vertex where

$Characteristic\_Diff = 0$. 2) One of these triangles is crossed by the outer contour.

To make the outer snake contracts and the inner ones expand we assign an inward normal force like in the equation 21 to the outer one and an outward normal force for the inner one.

As we are using the entropy condition, we can assign a fixed (interior) point to each inner snake. When the outer snake splits in two or more snakes these points can be used to easily identify which inner snakes are enclosed by an outer one. The freezing point is used to indicate that a T-snake found an equilibrium position (we say also that it is *frozen*).

Figure 9: Initialization of the Dual_t_Snakes method.

In the following algorithm we call *dual snakes* a list of T-snakes such that the first one is an outer contour and the next ones are inner contours. The algorithm can be summarized as follows:

Initialization: Put all the dual snakes into a queue. Until the queue is empty do:

1. Pop out a dual snakes from the queue.
2. Use the energies (equations 24 and 25) and the affinity function to decide the snake to process.
3. If the snake motion remains bellow some threshold increase the normal force until the snake energy starts decreasing.
4. Remove the added normal force liberating it until the temperature of all snaxels falls bellow the freezing point.
5. Analyze the $Characteristic\_Diff$ to identify if the snake being processed is close to an snake of the other type (inner/outer). In this case, remove the dual snake from the queue and go to step 1. Otherwise, mount the resulting dual snake(s) and go to the beginning step 0.

When an outer snake splits, each outer snake generated will be part of a new dual snakes. Each dual snakes is completely independent of the others. So, all the dual snakes on the queue can be processed in parallel. The result obtained by the Dual-T-Snakes method is in general two snakes bounded each object boundary (see Figure **??** on section 9 as an example). We will take advantage of such reduction of the search space later.

The normal force defined by equation (21) is not efficient for noise images. In this case, the following definition could improve the performance of the algorithm:

$$F_i(v_i) = Val(v_i)\, kn_i, \tag{26}$$

where:

$$\begin{aligned} Val(v_i) &= 1 \text{ se } I(v_i) < T, && (27)\\ Val(v_i) &= 0 \text{ otherwise}, && (28) \end{aligned}$$

Just as in [18] we control the evolution of the T-Snakes by ensuring that a contour can not move more than one pixel, in one interaction. Such policy guarantees the stability and consideration of all appropriate image data.

The extention of this algoruthm to 3D can be accomplished by using the T-Surfaces model of section 5. The *Dual-T-Surfaces* algorithm have the same steps above. Also, the CF triangulation used in the T-Surface model can be used to compare positions of the surfaces through an analogous criterium to that one given by conditions 1) and 2) above.

7 Segmentation Framework

After the Dual-T-Snakes algorithm stops we can take advantage of the reduction in the search space provided by it, and then apply a DP snake model to obtain the boundary enclosed by each dual snakes. In this way we take advantage of both the topological abilities of the Dual-T-Snake and the global minimization properties of the DP (robustness to noise and the guarantee that the global minimum is achieved).

As each boundary is enclosed by a dual snakes (Figure **??**) the Viterbi algorithm [19] is a suitable dynamic programming method. In this algorithm the search space is constructed by discretizing each curve in N points and establishing a matching between them (Figure 10.a). These points are then connected by segments which are subdivided in M points to provide a $discrete\ search\ space$ with NM points.

As we are looking for closed contours the solution adopted is to perform two searches [19]. In the first one two adjacent segments in the search space (Figure 10.a) are chosen and an open contour search is done. In the next search, two points are chosen lying in the middle of the solution obtained in the first step (Figure 10.b). These points are used as the start and end points for the new search which gives

Figure 10: DP for closed contours (a)First search (b)Final solution

the border we seek. The energy functional used is given by [19]:

$$E_s = \sum_{i=0}^{N-3} E_i, \qquad (29)$$

where: $E_i = \alpha E_{int}\left(v_i, v_{i+1}, v_{i+2}\right) + \beta E_{ext}\left(v_{i+1}\right) + \lambda E_{line}$, were α, β and λ are parameters to be defined in advace and: $E_{int}\left(v_i, v_{i+1}, v_{i+2}\right) = \frac{1}{2}\left(v_i - 2v_{i+1} + v_{i+2}\right)^2$,
and:
$E_{ext}\left(v_i\right) = -\left\|\nabla I\left(v_i\right)\right\|^2$ and $E_{line}\left(v_i\right) = -\left|I\left(v_i\right)\right|$.

The corresponding recurrence relation is the classical one given by [2]:

$$S_{i+1}\left(v_{i+2}, v_{i+1}\right) = \min_{v_i} \left\{S_i\left(v_{i+1}, v_i\right) + E_i\right\}, \qquad (30)$$

where E_i was defined just above.

In the second search we can use the search space already defined or a more appropriate one.

The Dual-T-Snakes method plus DP result in a boundary extraction procedure composed of four steps: (a) The user defines seed points inside the objects and initializes the outer snake; (b) Computation of the Affinity operator and affinity function; (c) Application of the Dual-T-Snake Algorithm; d) Find the final boundaries using Dynamic Programming.

This framework inherits the capability of changing topology from the T-Snakes which allows its application in the case where the objects of interest can be represented by connected closed regions immersed in a background. Another advantage is the ability to reject local minima of the Dual-T-Snakes due to the energy/affinity and a proximity criterium. Finally, the reduction of the search space allows to avoid the problems caused by memory and storage requirements of the dynamic programming.

8 Initializing Snakes by PL Methods

The initialization method that we propose in [15, 11] is based on some scale and topological constraints for the object of interest.

At first, we assume that the intensity (grey level) patterns of an object O (or of the background) can be characterized by a threshold T or some statistics (mean μ and variance σ) of the image field I [33]. That means:

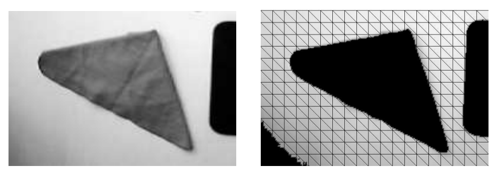
$$p \in O \Longrightarrow I\left(p\right) > T or \left|I\left(p\right) - \mu\right| \leq \sigma \qquad (31)$$

If the image field is not appropriated or we have a multivalued image (like colors and textured images), any other field derived from the image could be used [22].

Now, we assume a *Local Scale Property*: Given a point $p \in O$, let r_p be the radius of a hyperball B_p which contains p and lies entirely in the object region. We assume that $r > 1$.

This local scale property implies that we can reduce the resolution of the image without losing the objects of interest.

This is a basic idea for this work which is pictured on Figure 11.a. In this simple example we have a threshold which identifies the object ($T > 150$). In the Figure 11.b we have a CF triangulation whose grid resolution is 10×10, and the field thresholded.



(a) (b)

Figure 11: (a)Original image. (b)Grid and tresholded field.

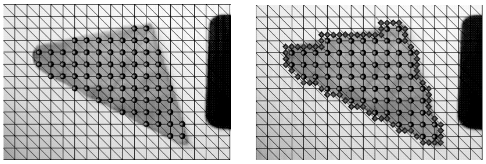
Now, we can define a simple function, called a *Object Characteristic Function*, as follows:

$$\chi : D \subset \Re^3 \rightarrow \{0, 1\} \qquad (32)$$

such that $\chi\left(p\right) = 1$ if $p \in O$ and $\chi\left(p\right) = 0$ and otherwise, where p is a node of the triangulation.

We can do a step further, shown in Figure 12.b, where we present a curve which belongs to the triangles in which the characteristic function (marked nodes) changes its value. Observe that this curve approximates the boundary we seek.

The generation of that curve (or surface for 3D) is a process which fits well in the subject of piecewise linear approximation of implicitly defined objects whose basic concepts was presented on section 4.1.1.



(a) (b)

Figure 12: (a)Characteristic Function. (b)Boundary approximation.

8.1 Our Method

The initialization method proposed in [15, 11] has the following steps: 1) Extract statistics of the objects or the background; 2) Triangulation of the domain; 3) Define an *Object Characteristic Function* as follows; 4) PL Manifolds generation; 5) Apply T-Surfaces (or T-Snake) model.

The region based statistics can be extracted by image histograms, or pattern recognition techniques [21].

The steps (2)-(5) rely on the following constraints supposed for the objects boundary: *a) Minimum area for the interior region, b) Closedness, c) Orientedness, d) Connectedness*.

The triangulation has a grid whose resolution is application dependent. However, an important point of our method is its multiresolution/multigrid nature: having resolved (segmented) the image in a coarser (grid) resolution we can detect regions where the grid has to be refined and then apply the method again only in these regions.

The detection of such regions may be done by using prior knowledge of parameters like area and symmetries [38] expected by the objects.

The PL Manifolds desired can be determined by the algorithm of section 4.1.2 for 2D images, or the alghorithm of section 5 for 3D data.

Among the PL Manifolds generated, a simple inspection would discard those which do not verify (a). For the remaining curves, those which do not satisfy (b) can be easily discarded by noticing that they start and end on the image frontiers (right side region on Figure 11).

To satisfy d) we have to discard curves interior to a larger one. As the remaining curves are closed, when finding a traverse triangle, the PL Manifold to be generated is assumed to represent an object. For example, the internal curves corresponding to the white regions inside the curve shown in Figure 13.a are discarded automatically when the PL Manifold shown in white is found.

For 2D images, the properties 1 and 2 of section 4.1.2 assure that the remaining curves limit regions with the properties (a),(b) and (d) above as well as that they are oriented.

For 3D, the orientation of the PL manifolds can be proved rigorously by applying the definition of the Characteristic function given by expression 16 and the theory of

(a) (b)

Figure 13: (a)PL Manifold. (b)Final result of the method.

orientation for PL Manifolds [1].

Our methodology (triangulation,threshold and object characteristic function) shares the basic elements used in methods based on iso-surface generation in hierarchical 3D meshes [17]. But, these methods in general do not use restrictions (a)$-$(d) which formalize a *prior knowledge* of the structures of interest. Also, we aim to take advantage of the simplicity of a CF triangulation.

The complexity of the procedure is $O(N)$ where N is the number of triangles.

Finally, we can use T-Snakes (or T-Surfaces) model to improve the approximations for the desired boundaries.

9 Experimental Results

In this section we present some results for 2D syntetical and medical images, concerning to the segmentation framework (section 7) and the initialization method of section 8.

First of all, we demonstrate the eficience of the Dual-T-Snakes to avoid local minima and its topological abilities for images with noise in the background.

The first point is the parameter choice for the Dual-T-Snakes model. As we have two snakes to initialize it is importante to reduce the number of parameters in order to make easier the application of the method. Thus, we use only rigidity, normal and external forces. The evolution equation for the Dual-T-Snakes used in the tests bellow is so given by:

$$v_i^{(t+\Delta t)} = v_i^t + \frac{\Delta t}{\gamma_i}\left(\beta_i^t + F_i^t + f_i^t\right). \qquad (33)$$

We end this section applying the method of section 8 for medical images. In the Appendix A we present a result of the multiseed fuzzy connectedness segmentation method and indicate how to integration this method with the Dual-T-Snakes approach.

9.1 Topological Changes

In the following examples the focus is the topological capabilities of the Dual-T-Snakes. We start with three objets embeded in a background with gaussian noise of mean 40.0 and variance 0.2. The correspondig image and the Dual-T-Snakes initialization are on Figure 14.

(a) (b)

(c)

Figure 14: (a)Image to be processed. (b)Triangular grid used with resolution (3×3).(c)Dual-T-Snakes initialization.

The Figure 15 shows some steps of the Dual-T-Snakes evolution.

It is important to consider the situation if we have only one T-Snake. The Figure 15.b shows the outer T-Snakes frozen. It is possible to apply the definition (21) to take the snake away from this position. But, how to know automatically that the snake position is away from the boundary?

In the Dual-T-Snakes, the algorithm firstly identilfies that the proximity criterium of section 6 is not satisfied. This implies that at least one of the snakes is not over the desired boundary. So, the snake with larger energy (equations (24)-(25)) has to be processed.

The observed defects are due to the stop criterium used. The values of parameters used on equations (20),(21) and (26) are the following: $k = 200.0$, $\lambda = 50.1$, $b = 50.0$, $T = 44.0$.

The total number of interactions for the outer snake to reach the final position of Figure 15.f was 49. The interac-

(a) (b)

(c) (d)

(e) (f)

Figure 15: (a) Interaction 15. (b) Outer snake frozen on interaction 25. (c) Snake inside triangular object is frozen. (d) Split of outer snake. (e) New split and Dual Snakes resulting. (f) Final Solution.

tions for the inner snakes summed 78. These are considered satisfatory.

In the Figure 16, we show a sequence of results obtained for the images which noise are indicated on Table 1. The grid resolution is 5×5. We show also the Dual-T-Snakes solution projected on the grid to allow a better qualitative comparison.

An important point arrives when considering topological changes: a set of parameters may be optimum for one object but may not be optimum for another one. This is due to the presence of poins with higher curvature or non uniform intensiy patterns.

One possibility to correct this problem could be increasing the image force parameter to get more resistence to the snake evolution near the bondaries. Certanly such policy makes the snake more sensitive to local minima, but it is not a problem in the Dual-T-Snakes framewor. For example, we have to set $\lambda = 100.1$ (instead of 50.1 above) to obtain the result on Figure 16.b.

Image	Mean , Variance
Figure 16.a	$\mu = 40.0$, $\sigma = 1.0$
Figure 16.c	$\mu = 40.0$, $\sigma = 4.0$
Figure 16.e	$\mu = 40.0$, $\sigma = 20.0$
Figure 16.g	$\mu = 40.0$, $\sigma = 30.0$

Table 1: Images used for tests of the Dual-T-Snakes pictured on Figure 16.

The parameters values used for the Dual-T-Snakes are shown on Table 2.

Image	k	λ	b	T	Freezing P.
Figure 16.a	200.0	100.1	10.0	44.0	10
Figure 16.c	150.0	100.1	10.0	44.0	10
Figure 16.e	200.0	100.1	20.0	44.0	10
Figure 16.g	200.0	100.1	10.0	44.0	10

Table 2: Parameters for Dual-T-Snakes corresponding to the results shown on Figure 16. The grid is 5×5.

The Figure 17 shows a typical result obtained through the Viterbi algorithm. The parameters used in equation (29) are the following: $\alpha = 1200.0, \beta = 0.5$ and $\lambda = 140.0$

Despite of the simplicity of the shapes in these examples, they characterizes a typical situation where the Dual-T-Snakes shows its capabilities: the interior and the exterior of the object of interest imposes dificulties to the snake evolution.

Another important point is that the Dual-T-Snakes fits well with multiscale methods like the Gaussian one. These methods reduce noise but also blurring the boundaries which

(a) (b)

(c) (d)

(e) (f)

(g) (h)

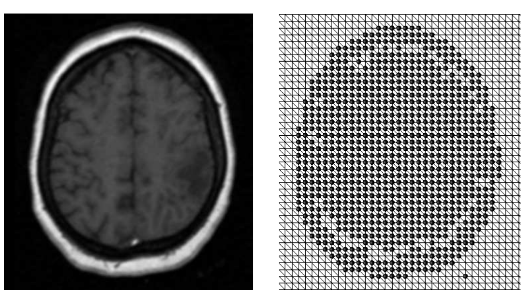
Figure 16: Dual-T-Snakes solution and corresponding projection oh grid 5×5.

Figure 17: Dynamic programming solution.

is a desired effect for Dual-T-Snakes, as the method is used mainly to limitate the region of interest.

9.2 Medical Images

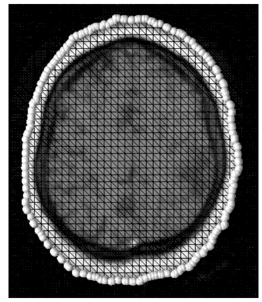
The Figure 19 shows the result obtained by steps (1)-(4) of the initialization method of section 8 corresponding to Figure 18. This figure is a particular slice from a set of 79 ones. We have applied our method for all of then with grid resolution of 5×5 and threshold $T = 5.0$. Just as solution pictured on Figure 19, the obtained results are rough approximations of the geometry of the head bone.



(a) (b)

Figure 18: (a)Original image. (b)Triangulation and Object Characteristic Function.

Now, we can use this result to initialize automatically the Dual-T-Snakes model for extracting the region corresponding to the head bone. It is just a matter of taking an off-set of the curve pictured on Figure 19.a. The off-set will



(a) (b)

Figure 19: (a)Solution of steps (1)-(4). (b) Initialization of the Dual-T-Snakes.

be the inner snake and the curve obtained in the initialization procedure the outer one. The Figure 19.b shows the Dual-T-Snakes initialization.

The corresponding solution and the final boundary obtained (Vitery solution) are pictured Figure 20.(e)-(f). The Figures 20.(a)-(d) shows the Dual-T-Snakes and the Viterbi result for slices 15 and 55 of the same set.

The problems observed in the solution pictured on Figure 20.f (botton) are due to non homogeneities on the intensity patterns inside the bone. We have to correct these problems in further works.

9.3 Cell Images

It is important to stress that the method described on section 8 gives in fact a rough segmentation of the image. As an example, consider the cell images on Figure 21.

The Figure 22 shows a result obtained by applying the method.

Despite of that there are regions where the resolution used 5×5 was not enough to segment the objects (interior of T-Snakes whose node elements are bigger ones), the *recognition* of the cells are almost completed.

That is the key idea of designing a method based on that one of section 8 for segmentation of images with target objects given by connected regions immersed in a background [13].

10 Future Works

10.1 Neural Networks for Snakes

The initialization of parametric deformable models used for handwritten digit recognition was addressed in [43] through neural networks. The targets of the initialization process in that case were the location of the control points (eight, in general) that define the B-spline model used for each digit template. The network was trained with backpropagation -

(a) (b)

(c) (d)

(e) (f)

Figure 20: Dual-T-Snakes solution and corresponding Viterby solution for slices 15, 55 and 60.

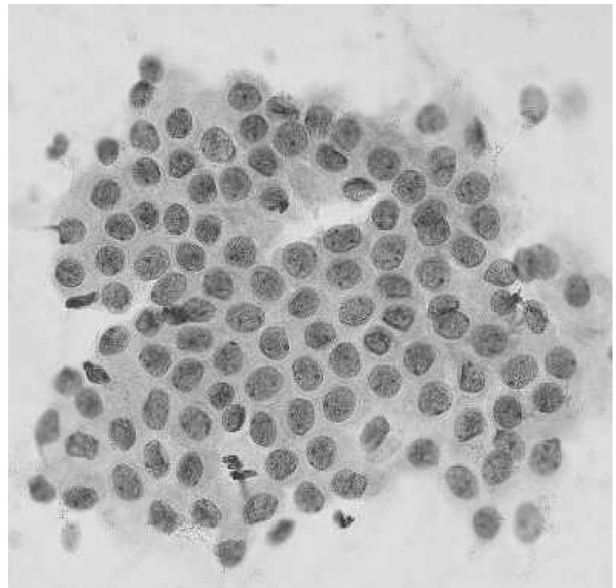


Figure 21: Endocervical Columnar cells.

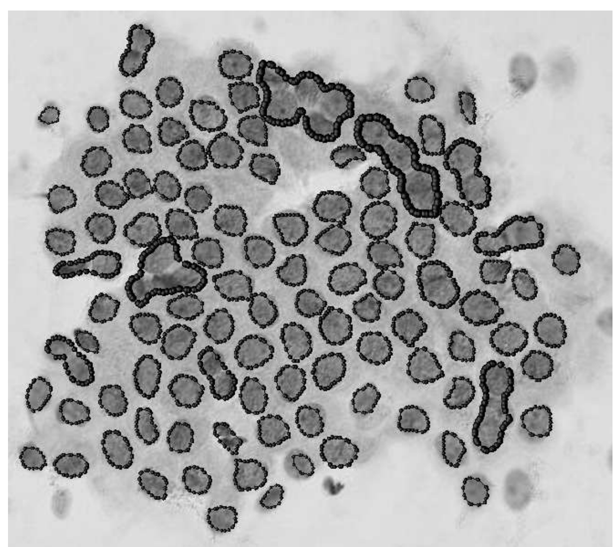


Figure 22: Solution for resolution 5×5.

a supervised trainning algorithm - to minimize the square error [43].

For using neural nets to initialize snakes, which are free form deformable models, the first point to be considered is that a neural net based on an unsupervised trainning algorithm could be more efficient than a supervise one, as we do not have a parameterized template of a specific structure. Among the possibilities, the Active Kohonen Nets (*AKN*) seems to be a simple one [24, 42].

The result of the *AKN* network is a *sketch* of the contour which means that the desired boundary lies in a region surrounding that sketch. Thus, if the topology of the boundary was preserved, we could made too off-sets of the Kohonen result to initialize the Dual-T-Snakes or the Viterbi algorithm if the search space was limited enough.

For noise images this approach generates artifacts in the background. Besides, the computational complexity of this neural net depends on the a number of neurons of the network and the number of pixels sent to it. These are technical aspects to be considered in further investigations.

10.2 Unifying Snakes and Live Wire

Live Wire is a user steered technique for boundary extraction. For 2D image it works by first initializing a point (*seed*) on the boundary using the cursor of the mouse and to any subsequent position of the cursor in the image, a minimum-cost path is computed via Dijkstra's algorithm [10]. A "live wire" is displayed in real time from the selected point to any subsequent position taken by the cursor. If the cursor is close to the boundary, the live wire snaps on to the boundary. For a complex boundary or when there is others strong but uninteresting object boundaries, it is necessary a set of live-wire segments. However, the result obtained is more accurate than a hand-drawn boundary. Thus, an interesting possibility is to use the live wire result to initialize snake models [28].

The research for the integration of live wire and snakes has just started and the optimal policies are not completely clear yet. Besides, there are not references in such works for 3D.

10.3 Tracking and Topological Changes

Tracking moving objects by snakes is a well known application for snake models [36]. However the methods in the literature are not convenient for tracking objects which undergo splits and merges during their evolution in a video sequence without adding extra machinery.

An interesting example of such a problem is the tracking of bubles evolving in a water reservoir.

For example, the velocity snakes is an active contour for real time tracking in combined spatio-velocity space. This technique is used for boundary tracking and motion analysis of non-rigid objects. However, the topology is supposed to be always the same during the motion.

Level Sets, on the other hand, incorporates topological changes in snake models but is not formulated in the spatio-velocity space. The solution obtained is a steady state one [35]. It is basically a geometric approach in which the time is an auxiliar variable.

It seemes for us that the challenge is to search for an approach which could unify these two methods. Such methodology could be used not only for tracking but also for geometric modeling of soft objects in interaction which would be another interesting application of it.

10.4 Snake System

Few works are reported in the literature about snake systems. In [20] we find an example of such a system. It is a semi-automatic system which combines motion estimation techniques (block matching and optical flow) with snakes. The G-Snake Model - a snake model which combines a *shape matrix* that contains the prior knowledge of the shape of the objects of interest, and a Markov random field to model local deformation - has also a system whose code is open [25].

Recently, Lian at. al [27] also report an interative medical image segmentation system based on live wire paradigm and the *United Snakes*, which unifies several snake variants, including finite difference, B-Spline and Hermite polynomial snakes, within the framework of a general finite element formulation. The implementatio is done in Java. Another system reported in [5] also uses Java to implementa a kind of B-spline snake.

These examples have the disadvantage of do not be user-extendable. Such is a fundamental feature of the Modular Visualization Environmemnts [6]. These systems allows one chooses modular blocks which perform certain functions, from a library of (user-extendable) routines, and links these together in an interative visual programming environment. The Khoros system, for example, has already implemented many facilities for image processing as well as for software integration and development.

The integration of a new module - a *Snake Toolbox* - in such a system coud be very usefull not only for taking advantage of their software development facilities but also to give them another tool for segmentation

11 Conclusions

Snake models offer an attactive approach for geometry recovery and tracking because these models are able to represent complex and broad shapes variability, particularly in the context of medical imaging.

The strong sensitivity to the initial contour position and the topological limitations of the traditional snake model was addressed by many works. Among them, the Dual-T-Snake approach incorporates the topological capabilities of the T-snake and the capability of avoiding local minima of the Dual Active Contour approach in one unified method. One advantage of the Dual-T-Snakes is the reduction of the search space, allowing a dynamic programming technique (Viterby algorithm) to be applied more eficiently as we demonstrate in the experimental results.

An effincient procedure to initialize snake models closer to the boundaries was also presented when some topological and scale constraints can be assumed. In the future works we indicate some challenges in the field of snake models concerned to the initialization, tracking and software development for snakes.

12 Appendix A

In this section we outline the basic elements of a fuzzy segmentation method which starts from a set of seed points over the image. The method summarized bellow is based on the work [41]. Also, it has the advantage of incorporating the image information in only one field σ. Besides, it can be applied for collors images and textures [22].

For a positive integer M and the domain D of the image, a *M-semisegmentation* of I is a function σ which maps each $c \in D$ into a vector $\sigma^c = (\sigma_0^c, \sigma_1^c, ..., \sigma_M^c)$, such that $\sigma_0^c \in [0,1]$, for all values of m we have $\sigma_m^c = 0$ or σ_0^c, and $\sigma_0^c = \max\{\sigma_i^c; i = 1, ..., M\}$.

To let these definitions more clear for our context, let's suppose two seed point ($M = 2$) in the image. If they belong to different objects, say O_1 and O_2 the field σ should be such that $\sigma_1^c > 0$ and $\sigma_2^c = 0$ if $c \in O_1$ (analogously for $c \in O_2$). So, according to the definition above we would define $\sigma_0^c = \max\{\sigma_i^c; i = 1, 2\}$. If c belongs to the background the ideal situation would be $\sigma_1^c = \sigma_2^c = 0$, but in practice we have to treshold σ_0 to distinguish between objects and background.

The multiseed fuzzy connectedness segmentation is a method to find the field σ which is based on a notion of an affinity vector $\Psi = (\psi_1, ..., \psi_M)$ whose component ψ_i gives the likelihood of a pixel to belong to the object $O_i; i = 1, ..., M$. Let's see some other definitions.

A *fuzzy affinity on* D is a function $\psi : D^2 \to [0,1]$. We think of (c,d) as a link and $\psi(c,d)$ as its $\psi-$*strength.* We define a chain in $U \subseteq D$ from $c^{(0)}$ to $c^{(K)}$ to be a sequence $\left\langle c^{(0)}, ..., c^{(K)} \right\rangle$ of points in U and the $\psi-$*strength* of this chain as the $\psi-$*strength* of its weakest link $\left(c^{(k-1)}, c^{(k)}\right)$, $1 \le k \le K$. If $K = 0$ then the $\psi-$*strength* is defined to be 1.

For segmentation of multiple objects we should have one fuzzy affinity for object, which leads to a vector field $\Psi = (\psi_1, ..., \psi_M)$ on D^2. For $U \subseteq D$, $W \subseteq D$ and $c \in D$, we denotate $\mu_{i,U,W}(c)$ the maximal ψ_i-*strength* of a chain in U from a pixel in W. The value of $\sigma_i^c, i = 1, ..M$ is such that $\sigma_i^c = \mu_{i,D,V_i}(c)$, where V_i holds the seed points for the object O_i. A dynamic programming method is used to obtain $\mu_{i,D,V_i}(c)$ [41].

The affinity function we have used is definied as follows:

$$\psi_i(c,d) = \mu_\alpha(c,d)\left[\omega_1 g_1(I(c), I(d)) + \omega_2 g_2(F(c), F(d))\right], \quad (34)$$

$$g_1 = \exp\left\{-1/2\left[(1/2)(I(c) + I(d)) - m_{1i}\right]^2 / \sigma_{1i}^2\right\}, \quad (35)$$

$$g_2 = \exp\left\{-1/2\left[|F(c) - F(d)| - m_{2i}\right]^2 / \sigma_{2i}^2\right\}, \quad (36)$$

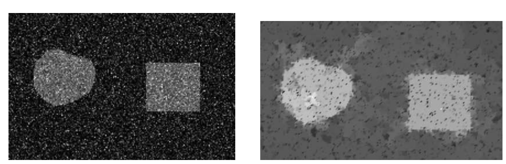
where I is the image intensity (grey-level), F is the image gradient, $(m_{1i}, \sigma_{1i}), (m_{2i}, \sigma_{2i})$ are the intensity mean and standard deviation of the image and its gradient over a neighbourhood around the seed points. The function $\mu_\alpha(c,d)$ is such that $\mu_\alpha(c,d) = 1$ if $|c - d| = 1$ and 0 otherwise (4-connected pixels).

12.1 Fuzzy connectedness Applications

In this section we show an example of the multiseed fuzzy connectedness segmentation method described above. Also, we discuss how to incorporates the result of this method in the Dual-T-Snakes approach.

The point is that the former needs at least one seed point inside each object of interest which can be used to initialize the inner snakes for the Dual-T-Snakes. Also, the field σ given by fuzzy connectedness can be used to define the normal signal when the image intensity is not proper to do this. These are the reasons of proposing to use multiseed fuzzy connectedness in our work.

Figures 23.a and 23.a show the test image and the segmentation obtained.



(a) (b)

Figure 23: (a)Original image. (b)multiseed fuzzy connectedness segmentation.

The normal signal in equation (21) can be defined for the outer snake as follows: where $sign(v_i) = -1$ if $\sigma(v_i) > T$ and $sign(v_i) = 1$ otherwise (analogously for the inner snakes); where T is a treshold for the field σ.

At first, we aims to use the field σ only for normal definition. The image force in equation (22) is calculated from the origial image to consider the image information more properly. These are further directions of our work.

13 Acknowledgments

We would like to acknowledge CNPq for the financial support for this work. Also we acknowledge Dr. A. Falcão from State Univ. of Campinas and his student E. Rodrigues for their helpful comments and suggestions.

References

[1] E. L. Allgower and K. Georg. *Numerical Continuation Methods: An Introduction*. Springer-Verlag Berlin Heidelberg, 1990.

[2] A. A. Amini, T. E. Weymouth, and R. C. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Trans. on Pattern Analysis and Machine Intel.*, 12(9):855–867, September 1990.

[3] P. Bamford and B. Lovell. A two-stage scene segmentation scheme for the automatic collection of cervical cell images. In *Proceedings of TENCON '97, Brisbane, Australia.*, December 1997.

[4] A. Black and A. Yuille, editors. *Active Vision*. MIT Press, 1993.

[5] P. Brigger, R. Engel, and M. Unser. B-spline snakes and java interface: An intuitive tool for general contour outlining. Technical report, Biomedical Imaging Group, Swiss Fed. Inst. of Tech.

[6] G. Cameron. Modular visualization environments: Past, present, and future (special focus). *IEEE Computer Graphics*, May 1995.

[7] L. D. Cohen. On active contour models and balloons. *CVGIP:Image Understanding*, 53(2):211–218, March 1991.

[8] L.D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Trans. on Pattern Analysis and Machine Intel.*, 15(11), November 1993.

[9] R. Durikovic, K. Kaneda, and H. Yamashita. Dynamic contour: a texture approach and contour operations. *The Visual Computer*, 11:277–289, 1995.

[10] Alexandre X. Falco, J.K. Udupa, S. Samarazekera, and B. E. Hirsch. User-steered image boundary segmentation. *Proceedings of SPIE on Medical Imaging, Newport Beach, CA*, 1996.

[11] A. Apolinario A. F. Oliveira G. Giraldi, E. Strauss. An initialization method for deformable models. In *Computer Graphics and Imaging (CGIM'2000 (to appear))*, 2000.

[12] G. A. Giraldi, L. M. Gonçalves, and A. F. Oliveira. Dual topologically adaptable snakes. In *Proceedings of the Fifth Joint Conference on Information Sciences (JCIS'2000, Vol. 2) - Third International Conference on Computer Vision, Pattern Recognition, and Image Processing*, pages 103–106, 2000.

[13] G. A. Giraldi, E. Strauss, and A. F. Oliveira. A boundary extraction approach based on multi-resolution methods and the t-snakes framework. In *International Symposium on Computer Graphics, Image Processing and Vision (SIBGRAPI'2000 (to appear)*, 2000.

[14] G. A. Giraldi, E. Strauss, and A. F. Oliveira. A boundary extraction method based on dual-t-snakes and dynamic programming. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2000)*, 2000.

[15] G. A. Giraldi, E. Strauss, and A. F. Oliveira. An initialization method for active contour models. In *Proceedings of the 2000 International Conference on Imaging Science, Systems, and Technology (CISST'2000)*, 2000.

[16] L.V. Gool, T. Moons, E. Powrls, and A. Oosterlinck. Vision and lie's approach to invariance. *Image and Vision Computing*, 13(4):259–277, 1995.

[17] R. Grosso and T. Ertl. Progressive iso-surface extraction from hierarchical 3d meshes. *Computer Graphics Forum (Eurographics'98 Conference Issue)*, 17(3):125–135, September 1998.

[18] S. R. Gunn and M. S. Nixon. A robust snake implementation; a dual active contour. *IEEE Trans. Pattern Anal. Mach. Intell*, 19(1):63–68, January 1997.

[19] Steve R. Gunn. *Dual Active Contour Models for Image Feature Extraction*. PhD thesis, Faculty of Engineering and Applied Science, Department of Eletronics and Computer Science., May 1996.

[20] M. Hoch and P. C. Litwinowicz. A semi-automatic system for edge tracking with snakes. *The Visual Computer*, 11:277–289, 1995.

[21] A. K. Jain, R. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):4–37, January 2000.

[22] Timothy N. Jones and Dimitris N. Metaxas. Image segmentation based on the integration of pixel affinity and deformable models. In *Proceedings of CVPR'98*, June, 1998.

[23] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[24] T. Kohonen. *Self-organizing and associative memory*. Springer, 3nd edition, 1989.

[25] Kok F. Lai and R. T. Chin. Deformable contours: Modeling and extraction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(11):1084–1090, November 1995.

[26] F. Leymarie and M. D. Levine. Tracking deformable objects in the plane using and active contour model. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(6):617–634, June 1993.

[27] J. Liang, T. McInerney, and D. Terzopoulos. Interative medical image segmentation with united snakes. In *MICCAI'99*, 1999.

[28] J. Liang, T. McInerney, and D. Terzopoulos. United snakes. In *ICCV'99*, 1999.

[29] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[30] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(2):158–175, 1995.

[31] T. McInerney and D. Terzopoulos. Topologically adaptable snakes. In *Proc. Of the Fifth Int. Conf. On Computer Vision (ICCV'95), Cambridge, MA, USA*, pages 840–845, June 1995.

[32] T. McInerney and D. Terzopoulos. Medical image segmentation using topological adaptable surfaces. In *Proc. CVRMed'97*, March 1997.

[33] T. McInerney and D. Terzopoulos. Topology adaptive deformable surfaces for medical image volume segmentation. *IEEE Trans. on Medical Imaging*, 18(10):840–850, October 1999.

[34] T. J. McInerney. *Topologically Adaptable Deformable Models for Medical Image Analysis*. PhD thesis, Department of Computer Science, University of Toronto, 1997.

[35] N. Paragios and R. Deriche. Geodesic active contour and level sets for the detection and tracking of moving objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(3):266–280, 2000.

[36] N. Peterfreund. Robust tracking of position and velocity with kalman snakes. *IEEE Trans. on Pattern Analysis and Machine Intel.*, 21(6), June 1999.

[37] J. M. Carstensen R. Fisker. On parameter estimation in deformable models. In *14th International Conference on Pattern Recognition*, pages 762–766, August 16-20 1998.

[38] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer*, pages 20–27, July 1994.

[39] G. Storvik. A bayesian approach to dynamic contours through stochastic sampling and simulated annealing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(10):976–986, 1994.

[40] R. Szelinski, D. Tonnensen, and D. Terzopoulos. Modeling surfaces of arbitrary topology with dynamic particles. In *Proc. Conf. Computer Vision and Pattern Recognition(CVPR'93)*, pages 82–87, New York, NY, Los Alamitos CA, June 1993. IEEE Computer Society Press.

[41] J. Udupa and S. Samarasekera. Fuzzy connectedness and object definition: Theory, algorithms and applications in image segmentation. *Graphical Models, Image Processing*, 58(3), 1996.

[42] N. A. Vasconcelos. Mapas auto-organizativos e suas aplicaçẽs. Master's thesis, Departamento de Engenharia de Sistemas e Computação-COPPE/UFRJ, 2000.

[43] C. K. I. Williams, M. Revow, and G. E. Hinton. Instantiating deformable models with a neural net. *Computer Vision and Image Understanding*, 68(1):120–126, 1997.

[44] Gang Xu, E. Segawa, and S. Tsuji. Robust active contours with insensitive parameters. *Pattern Recognition*, 27(7):879–884, January 1994.