Mathematical Elements of Deformable Models for Image Processing

Gilson A. Giraldi
and
Paulo S. S. Rodrigues

LECTURE NOTES
NATIONAL LABORATORY FOR SCIENTIFIC COMPUTING

PETRÓPOLIS, RJ - BRASIL
SEPTEMBER 2009

To our Families

October 20, 2009

Contents

List of Figures

Chapter 1

Preface

Active Contour Models, also called Snake models, are powerful techniques for boundary extraction and segmentation of $2D$ images. Despite of their abilities, the non-invariance of the internal energy under affine transformations, the non-convexity of the model energy functional and the inability to deal with topological changes are known limitations for most of these methods. In this book we describe some techniques to address these limitations. The non-invariance of the internal energy has been addressed in the context of active shape models and Lie groups. The non-convexity of the model energy can be addressed through Dual contour approaches, diffusion approaches, as well as an automatic procedure to initialize the model closer to the desired boundary. The problem of topological changes is addressed by embedding the snake in the framework of a simplicial decomposition of the domain or through implicit formulations. In this text, we offer some background in parametric snakes, followed by a taxonomy of deformable models for image processing. Then, we discuss mathematical elements behind the main works that address the mentioned problems. Next, we survey snake approaches according to the presented taxonomy. In order to complete the material, we dedicate a Chapter for extensions to Deformable Surface approaches. Finally, this lecture ends with a discussion on snake approaches, some drawbacks and future works. We survey applications for shape recovery in cell and ultrasound images.

Chapter 2

Introduction

Deformable Models, which includes the popular snake models [1] and deformable surfaces [2, 3], are well known techniques for boundary extraction and tracking in $2D/3D$ images. Basically, these models can be classified in three categories: parametric, geodesic snakes and implicit models. The relationships between these models have been demonstrated in several works in the literature [4, 5].

Parametric Deformable Models consist of a curve (or surface) which can dynamically conform to object shapes in response to internal (elastic) forces and external forces (image and constraint ones) [6]. Snake models, also called active contour models, are $2D$ deformable models proposed by Kass at al. [1] which have been successfully applied in a variety of problems in computer vision and image analysis [6, 23]. Its mathematical formulation makes easier to integrate image data, an initial estimated, desired contour properties and knowledge-based constraints, in a single extraction process [6]. In what follows, we will focus mainly on snake models.

For Geodesic Snakes, the key idea is to construct the evolution of a contour as a geodesic computation. A special metric is proposed (based on the gradient of the image field) to let the state of the minimal energy corresponds to the desired boundary. This approach allows to address the parameterization dependence of parametric snake models and can be extended to 3D through the theory of minimal surfaces [7, 5].

Implicit models, such as the formulation of level set used in [8], consist of embedding the snake as the zero level set of a higher dimensional function and to

solve the corresponding equation of motion. Such methodologies are best suited for the recovery of objects with unknown topologies which is a limitation for most of the parametric models (see [9] for a review).

In fact, despite of the mentioned capabilities, parametric models in general can not deal with topological changes. Among the approaches to deal with the topological limitations of the traditional snake model [10, 11, 12, 13], the T-Snakes has the advantage of being a general one [14]. In this model, topological changes are addressed by embedding the snake in the framework of a simplicial domain decomposition, using classical results in the field of combinatorial topology [89].

Besides, parametric models are too sensitive to their initial conditions due to nonconvexity problems (see [118] and references therein). To address this problem, some authors have proposed Dual approaches [17], multiscale techniques [15], simulated annealing [90], dynamic programming (DP) [16], learning techniques for initialization [98, 105, 81], as well as a two stage approach [91, 42, 22]: (1) the region of interest is reduced; (2) a dynamic programming (DP) technique is used to find the object boundaries. Among these methods, dual approaches, also called **dual snakes** [17], are of special interest in this work. The basic idea of the dual snakes is to reject local minima by using two contours: one which contracts from outside the target and one which expands from inside. Such proposal makes possible to reduce the sensitivity to initialization through the comparison between the two contours energy and positions. 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 [17].

The non-invariance under affine transformations is another limitation of the traditional snake models. As a consequence, the internal energy is sensitive to distortions due to changes in viewing geometry. From a dynamical point of view, it means that the elastic forces may affect the efficiency of the energy minimization process [19, 20]. Some methods have been proposed to address this problem [19, 21], even in the context of dual active contour models.

In this lecture notes we describe some techniques to address these limitations. In Chapter 3, we offer the theoretical background on snake models. We start on section 3.2 with a review on the original snake model which is a continuous and parametric one. Then, in section 3.2.1, discrete models generated through finite difference approximations for the original model are described.

The Chapter 4 gives mathematical elements in invariance theory and Lie Groups approach, linear and non-linear scale space, Support Vector Machines (SVM) and Neural Nets. These methods have been used for invariant snakes methods [21], pre-processing techniques [15, 23] and model initialization [98, 105, 81], as will be discussed in Chapters 5-8.

The non-invariance under affine transformations is addressed in Chapter 5 through the application of invariance theory for snakes. We present some details of the AI-Snake - a discrete and affine-invariant snake model [19]. Then, we discuss the application of Lie Groups in this context. We analyze the effects of reparameterization and indicate potential applications of Lie's Invariance Theory for snakes (section 5.2.1).

Linear scale spaces are commonly defined by the Gaussian blurring. The standard deviation of the Gaussian controls the scale to process the data. In this case, we get a multiscale representation of an image by generating a one-parameter family of derived images through the convolution of the original image with the Gaussian kernel. Multiscale techniques have been developed to provide a way to isolate, analyse and interpret structures of different scales within an image. In the context of snake models, multiscale techniques can be used to parameterize the image field over the snake evolution, as we shall see in section 7.2.

Support Vector Machines (SVM) and multilayer perceptrons (MLP) are supervised statistical learning methods [101]. These methods cover important topics in classical statistics such as discriminant analysis, regression methods, and the density estimation problem [113]. Moreover, in the classification problem, we try to distinguish separated subsets (classes) and to find out an approach to automatically label data points according to the class they belong [101]. In the case of supervised methods, the learning typically occurs through training, or exposure to a know set of input/output data. The training algorithm iteratively adjusts model parameters, which store the knowledge necessary to solve specific problems. The key idea to apply these methods for snake initialization is to get a classifier (by training) that provides the initial contours which are close to the correct boundaries. Sections 6.4 and 8.3 present works in this area that uses SVM and MLP as the classifiers.

Automatic initialization of snakes and multiscale representations are approaches to address the non-convexity problem. Another way is through dual

snakes, which will be discussed during Chapters 5-9.

Section 3.3 offers a taxonomy in this area that steers the presentation. We sort the target techniques according to that taxonomy. Therefore, Chapters 5-8 are oriented following the proposed taxonomy. Besides, we survey applications for shape recovery in cell and ultrasound images. We also dedicate the Chapter 9 for extensions to Deformable Surface approaches. Finally, Chapter 10 presents some discussions and drawbacks of snake approaches.

Chapter 3

Background in Snake Models

3.1 Introduction

The snake model [1] is a kind of deformable model which can be used to process 2D or 3D data [6]. Considering as a functional energy minimization process then snake models consist 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 original one proposed by Kass at al. [1].

3.2 Original Snake Model

In [1] a snake is geometrically defined as a parametric contour c, here assumed to be closed, embedded in a domain $D \subset \Re^2$:

$$c : [0,1] \rightarrow D \subset \Re^2; c(s) = (x(s), y(s)). \tag{3.1}$$

We can define a deformable model as a space of admissible deformations (contours) Ad and a functional E to be minimized [44]. This functional represents the energy of the model and has the form:

$$E : Ad \rightarrow \Re,$$

$$E(c) = w_1 E_{ten}(c(s)) + w_2 E_{rig}(c(s)) + \gamma E_{ext}(c(s)); \tag{3.2}$$

where

$$E_{ten} = \int_{[0,1]} \left\| \frac{dc}{ds} \right\|^2 ds\ , \tag{3.3}$$

$$E_{rig} = \int_{[0,1]} \left\| \frac{d^2c}{ds^2} \right\|^2 ds\ , \tag{3.4}$$

$$E_{ext} = \int_{[0,1]} P(c(s))\, ds, \tag{3.5}$$

are the tensile, rigidity and external energy terms, respectively. The first two terms compose the **internal energy** of the model. The parameter w_1 (tension) gives the snake the behavior of resisting to the stretch while the parameter w_2 (rigidity) makes the snake less flexible and smoother. In order to verify these features let α represents a reparameterization; that is, $\alpha = \alpha(s)$. So, we can compute the derivatives:

$$\frac{dc}{ds} = \frac{dc}{d\alpha}\frac{d\alpha}{ds}, \tag{3.6}$$

$$\frac{d^2c}{ds^2} = K\overrightarrow{n}\left(\frac{d\alpha}{ds}\right)^2 + \overrightarrow{T}\frac{d^2\alpha}{ds^2}, \tag{3.7}$$

where $\overrightarrow{T}$ e $\overrightarrow{n}$ means the tangent and normal vectors and K the curvature. Therefore, we observe from equation (3.6) that if we increase the arc length ($d\alpha/ds > 1$) we also increase the value of the energy (3.3). Also, from equation (3.7), we see that the energy (3.4) incorporates the curvature K in agreement with the explanation for the parameter w_2. Besides, the internal energy has a global minimum when the curve shrinks to a point. So, we have a contraction effect which may be undesirable and difficult to control [45]. The problem behind this fact is the non-invariance of the internal energy under affine transformations which was discussed in [19, 21].

The parameter γ scales the external (image) energy and controls the sensitivity to image features. The parameters w_1, w_2 and γ can be constant or dependent on

s [15]. In the external energy E_{ext}, P is the potential associated with the image data and is narrow related with the features we seek. For edge detection in a grey scale image a possible definition is [6]:

$$P = -\left\|\nabla I\right\|^2, \tag{3.8}$$

or simply:

$$P = -\left\|\nabla I\right\|, \tag{3.9}$$

where I is the image intensity and ∇ is the gradient operator. When minimizing the functional $E(c)$ we get a curve localized in high gradient regions, due to energy E_{ext}, but with the desired smoothness which is controlled by the internal energies E_{ten} and E_{rig}.

In the reference [1] it is applied Euler-Lagrange equations in order to minimize the energy functional given by expression (3.2) [92]. That functional has the form:

$$E = \int_0^1 L(c, c', c'')\, ds. \tag{3.10}$$

where:

$$c' = \frac{dc}{ds},\ c'' = \frac{d^2c}{ds^2},\ \text{e}\ L(c, c', c'') = \omega_1 \left\|c'(s)\right\|^2 + \omega_2 \left\|c''(s)\right\|^2 + \gamma P(c), \tag{3.11}$$

So, the corresponding Euler-Lagrange equations are given by [92]:

$$\frac{\partial L}{\partial x} - \frac{\partial}{\partial s}\left(\frac{\partial L}{\partial x'}\right) + \frac{\partial^2}{\partial s^2}\left(\frac{\partial^2 L}{\partial x''}\right) = 0, \tag{3.12}$$

$$\frac{\partial L}{\partial y} - \frac{\partial}{\partial s}\left(\frac{\partial L}{\partial y'}\right) + \frac{\partial^2}{\partial s^2}\left(\frac{\partial^2 L}{\partial y''}\right) = 0, \tag{3.13}$$

or, in a compact shape:

$$\frac{\partial L}{\partial c} - \frac{\partial}{\partial s}\left(\frac{\partial L}{\partial c'}\right) + \frac{\partial^2}{\partial s^2}\left(\frac{\partial^2 L}{\partial c''}\right) = 0. \tag{3.14}$$

For the Lagrangian L in expression (3.2) we obtain the following equations:

$$-2\frac{d}{ds}\left(\omega_1 \frac{dc}{ds}\right) + 2\frac{d^2}{ds^2}\left(\omega_2 \frac{d^2c}{ds^2}\right) - \gamma \nabla P = 0. \tag{3.15}$$

The first and second terms gives the internal (elastic forces) and the term:

$$F_{ext} = -\nabla P = 0. \tag{3.16}$$

defines the external force field. From a mechanical viewpoint, this equation summarize the fact that the *potential energy* defined by the functional (3.2) has an extremum if the internal and the external force has a null sum.

The process of minimizing the functional given in (3.2) can be viewed from a pseudo-dynamic viewpoint by considering the deformable contour a time-varying curve:

$$c\left(s,t\right) = \left(x\left(s,t\right), y\left(s,t\right)\right), \tag{3.17}$$

and generalizing expression (3.15) by incorporating a *pseudo* time t:

$$\frac{\partial c}{\partial t} - w_1 \frac{\partial^2 c}{\partial s^2} + w_2 \frac{\partial^4 c}{\partial s^4} - \gamma \nabla P = 0, \tag{3.18}$$

where we are supposing w_1 and w_2 constants. In section 7.3 we return to this formulation and derive it through a variational approach for continuous mechanics.

The rigidity in expression (3.18) makes the snake too rigid and may be not efficient to get finer details. Therefore, as usual for dual snake models, we set $w_2 = 0$ in expression (3.18). The obtained model has the following evolution equation:

$$\frac{\partial c}{\partial t} - w_1 \frac{\partial^2 c}{\partial s^2} - \nabla P = 0. \tag{3.19}$$

3.2.1 Discrete Models

In order to solve the equation 3.18, or its simplified version (3.19), for an initial 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 [15, 3]. The discretization of expressions (3.2) and (3.19) through finite difference equations generates a variety of discrete snake models (finite element methods are considered in [3]). In these models, a (closed) snake is formulated as a list of points $v_0, v_2, ..., v_{N-1}$, also called **snaxels**, that defines a closed polygonal which represents the deformable

curve. The first order approximations for the derivatives $\partial c/\partial t$, $\partial c/\partial s$, $\partial^2 c/\partial s^2$ are, respectively, given by:

$$\frac{\partial c\left(s_i, t_j\right)}{\partial t} \approx \frac{v_i^{t+\Delta t} - v_i^t}{\Delta t}, \tag{3.20}$$

$$\frac{\partial c\left(s_i, t\right)}{\partial s} \approx \frac{v_{i+1}^t - v_i^t}{\Delta s_i}, \tag{3.21}$$

$$\frac{\partial^2 c\left(s_i, t\right)}{\partial s^2} \approx \frac{v_{i-1}^t - 2v_i^t + v_{i+1}^t}{\left(\Delta s_i\right)^2}, \tag{3.22}$$

where Δt and Δs_i are discrete steps in time and space, respectively.

The external energy, defined by equation (3.5) may be discretized by evaluating the potential (equation (3.8)) for the snaxels in the list $v_0, v_2, ..., v_{N-1}$ and replace the integral by the summation:

$$E_{ext} = -\sum_{i=0}^{N-1} \left\|\nabla I\left(v_i\right)\right\|^2 h_i, \tag{3.23}$$

where h_i is the integration step. Thus, using expressions (3.20)-(3.23) we obtain the following discrete version for the snake energy in expression (3.2):

$$E = w_1 \sum_{i=0}^{N-1} \left\| \frac{v_{i+1}^t - v_i^t}{\Delta s_i} \right\|^2 h_i + w_2 \sum_{i=0}^{N-1} \left\| \frac{v_{i-1}^t - 2v_i^t + v_{i+1}^t}{\left(\Delta s_i\right)^2} \right\|^2 h_i - \gamma \sum_{i=0}^{N-1} \left\|\nabla I\left(v_i\right)\right\|^2 h_i, \tag{3.24}$$

The discretization of the evolution equation (3.19) is similarly obtained:

$$\left(\frac{v_i^{t+\Delta t} - v_i^t}{\Delta t}\right) - w_1 \left(\frac{v_{i-1}^t - 2v_i^t + v_{i+1}^t}{\left(\Delta s_i\right)^2}\right) - \gamma \nabla P\left(v_i\right) = 0, \tag{3.25}$$

where the term:

$$B_i = w_1 \left(\frac{v_{i-1}^t - 2v_i^t + v_{i+1}^t}{\left(\Delta s_i\right)^2}\right), \tag{3.26}$$

define the so called tensile force, respectively. The snake comes to rest when the tensile and external forces balance, which implies that:

$$\left\| v_i^{t+\Delta t} - v_i^t \right\| \approx 0. \tag{3.27}$$

The force B_i, as well as the equation (3.22), can be geometrically interpreted through the Figure 3.1 which pictures three consecutive snake points v_{i-1}, v_i, v_{i+1} of the (discrete) snake. The vector V_1 with origin in the snaxel v_i and end in the midpoint of the segment $\overline{v_{i-1}v_{i+1}}$ is parallel to B_i. So, the force B_i pushes the snaxel v_i towards the configuration of minimum curvature. Therefore, the resulting curve may be smoother than the original one.

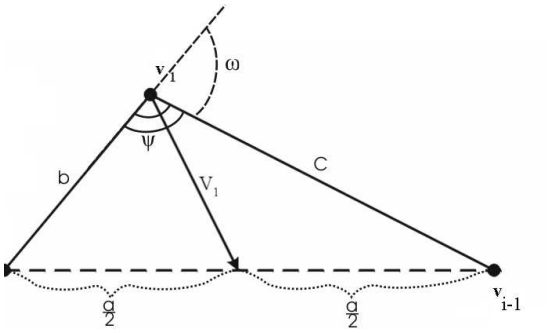


Figure 3.1: Geometric representation of the force B_i in expression (3.26) with vector V_1 parallel to B_i.

As already discussed before, the whole effect of the force defined by equation (3.26) is a tendency to contraction of the original curve. The Figure 3.1 helps to realize this fact in the discrete case. This may be a problem because if the image force field is not strong enough then the curve collapses to a point. This problem can be addressed by the balloon model [44] or through affine invariant models (see [19, 21] for details). Like in the continuous case, the external force given by expression (3.16) attracts the snake to strong edges in the image.

3.3 Taxonomy for Deformable Models

After the work of Kass at al. [1] a variety of models were proposed which taxonomy can be described as a tree, as observed in Figure 3.2. The first level of the tree shows the classification of $2D$ snake models, based on the curve representation,

that branches into two core classes: parametric and implicit. Then, each class roots its own subtree. The parametric models are classified according to the geometric model used which can be a discrete or continuous one. Finally, the deepest nodes show a classification according to the model evolution. Discrete models can be evolved through nonlinear optimization techniques, like steepest descent, or search based techniques (dynamic programming or greedy), or even through local deformations based on an evolution equation derived through local information over the deformable curve. Continuous snake models, like the original one described on section 3.2, can be evolved based on Euler-Lagrange equations or front propagation methods. Turning back to the top of Figure 3.2, we see the implicit models, formulated through level set theory [8, 32].

In Chapters 5-8, we review some examples of the classes just described. We focus on topological models (T-Snakes and Level Set), invariant and dual snakes, dynamic programming, greedy techniques, multiscale and learning methods for initialization. In order to provide the mathematical background, we discuss some topics in diffusion and invariant theory as well as statistical learning technique in Chapter 4.

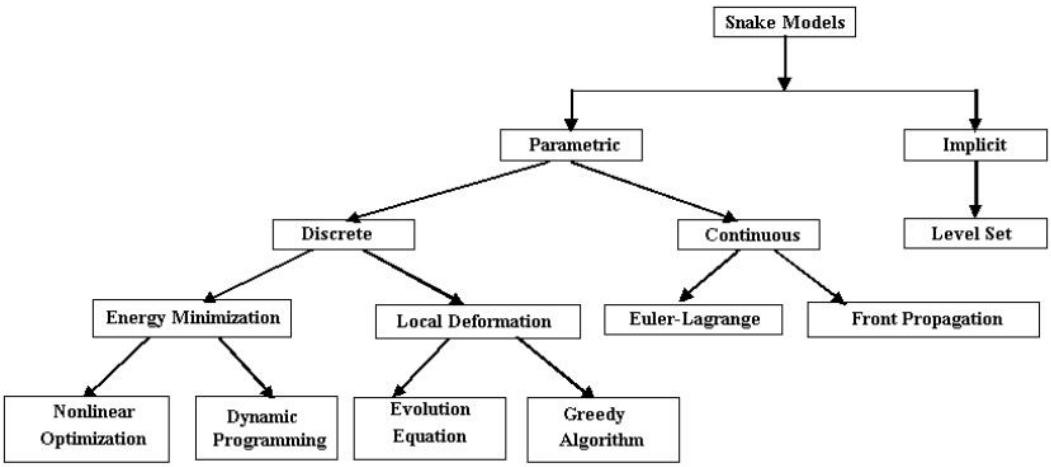


Figure 3.2: Taxonomy of deformable models for medical image segmentation

Chapter 4

Mathematica Elements

4.1 Introduction

In this chapter our aim is to offer some mathematical elements which are part of the current research in the field of snake models. Scale-space methods, invariance theory and learning models for snake initialization are the focus of this Chapter.

For noisy images the convergence of deformable models to the boundaries is poor due to the non-convexity of the image energy. This problem can be addressed through diffusion techniques [93, 15]. In image processing, the utilization of diffusion schemes is a common practice. Gaussian blurring is the most known one. Another approaches are the anisotropic diffusion [93] and the Gradient Vector Flow (GVF) [94]. From the viewpoint of deformable models, these methods can be used to improve the convergence to the desired boundary.

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 [44] and Gang Xu at al. [45] this problem is addressed by the addition of another internal force term to reduce the bad effects of the contraction force. 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 [95, 21]. This concept has been applied to 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 [17, 102].

The sensitivity to the initial contour position can also be addressed by a method which initializes automatically the snake closer to the boundaries [96]. An efficient methodology in this field would be worthwhile not only to save time calculation but also to facilitate the specification of parameters, a known problem for snake models [118]. Neural Networks and Support Vector Machines have been applied for initialization of deformable models [97, 81, 98].

Therefore, in the next section we cover topics in diffusion methods and GVF (section 4.2). In section section 4.3 we discuss invariance from the viewpoint of Lie groups and in section 4.4 we present statistical learning methods for snake initialization.

4.2 Diffusion Methods

Anisotropic diffusion is defined by the following general equation:

$$\frac{\partial I\left(x,y,t\right)}{\partial t} = div\left(c\left(x,y,t\right)\nabla I\right), \tag{4.1}$$

where I is a gray level image [93].

In this method, the blurring on parts with high gradient can be made much smaller than in the rest of the image. To show this property, we follow Perona and Malik [93]. Firstly, we suppose that the edge points are oriented in the x direction. Thus, equation (4.1) becomes:

$$\frac{\partial I\left(x,y,t\right)}{\partial t} = \frac{\partial}{\partial x}\left(c\left(x,y,t\right)I_x\left(x,y,t\right)\right). \tag{4.2}$$

If c is a function of the image gradient: $c\left(x,y,t\right) = g\left(I_x\left(x,y,t\right)\right)$, we can define $\phi\left(I_x\right) \equiv g\left(I_x\right)\cdot I_x$ and then rewrite equation (4.1) as:

$$I_t = \frac{\partial I}{\partial t} = \frac{\partial}{\partial x}\left(\phi\left(I_x\right)\right) = \phi'\left(I_x\right)\cdot I_{xx}. \tag{4.3}$$

We are interested in the time variation of the slope: $\frac{\partial I_x}{\partial t}$. Supposing that we can change the order of differentiation, a simple algebra is enough to demonstrate that:

$$\frac{\partial I_x}{\partial t} = \frac{\partial I_t}{\partial x} = \phi'' \cdot I_{xx}^2 + \phi' \cdot I_{xxx}.$$

At edge points we have $I_{xx} = 0$ and $I_{xxx} << 0$ as these points are local maxima of the image gradient intensity. Thus, there is a neighborhood of the edge point in which the derivative $\partial I_x / \partial t$ has sign opposite to $\phi'(I_x)$. If $\phi'(I_x) > 0$ the slope of the edge point decrease in time. Otherwise it increases, that means, border becomes sharper. So, the diffusion scheme given by equation (4.1) allows to blur small discontinuities and to enhance the stronger ones if the function $c(x, y, t)$ is properly chosen. For instance, in [93] authors suggest the following definition:

$$c = \left(\frac{1}{\left(1 + \left[\|\nabla I\| / K\right]^2\right)} \right). \tag{4.4}$$

If $c(x, y, t)$ is constant we get the traditional linear diffusion equation which fundamental solution corresponds to the Gaussian blurring. Levine at al. [15] applied hierarchical filtering methods, as well as a continuation method based on a (discrete) linear 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 (see section 7.2 also).

In the above scheme, I is a scalar field. For vector fields, a useful diffusion scheme is the Gradient Vector Flow (GVF). It was introduced in [94] and can be defined through the following reaction-diffusion equation [99]:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nabla \cdot (g \nabla u) + h(u - \nabla f), \\ u(x, 0) &= \nabla f \end{aligned} \tag{4.5}$$

where f is a function of the image gradient (for example, P in equation (3.8)), and $g(x), h(x)$ are nonnegative functions defined on the image domain.

The field obtained by solving the above equation is a smooth version of the original one which tends to be extended very far away from the object boundaries. When used as an external force for deformable models, it makes the methods less sensitive to initialization [94] and improves their convergence to the object boundaries. That is the basic idea behind GVF snakes, that means, this method replaces the external force, given by expression (3.16), by the force field of the GVF equation solution.

4.3 Lie Groups and Invariance

For the invariance theory to be useful we need a systematic way of finding invariants for a specific application. This requirement is satisfied by the Lie Groups approach.

The whole development could be summarized through the Lie Derivative. A complete presentation of this theory can be found in [100]. However, we are going to begin in a more practical description. We follow the steps found in [95].

Definition: A r-parametric **Lie Group** is a group G which is also a smooth differential manifold of dimension r such that the group operations [100]:

$$m : G \times G \longrightarrow G; \qquad m(g,h) = g \cdot h, \quad g,h \in G,$$

and inversion:

$$i : G \rightarrow G; \qquad i(g) = g^{-1}, \quad g \in G,$$

are smooth maps between differential manifolds.

Let us suppose that the quantities of interest are arranged in a column array $\overrightarrow{m} = (m_1, m_2, ..., m_{n_m})^T$. We will assume that there is a Lie group G, $n_p - parametric$, whose action transforms $\overrightarrow{m}$. Thus, G can be represented by a matrix $g = (g_{ij})_{n_m \times n_m}$, such that $g_{ij} = g_{ij}\left(p_1, p_2, ..., p_{n_p}\right)$ are smooth functions of the parameters $p_i \in \Re$, $i = 1, ..., n_p$. So, the group action can be represented by:

$$\overrightarrow{m}' = g \cdot \overrightarrow{m}, \tag{4.6}$$

where $\overrightarrow{m}'$ indicates the transformed vector (not a derivative).

In this context, an invariant f is a function $f(\overrightarrow{m})$ which does not change under the action of G, that is:

$$f(\overrightarrow{m}') = f(g \cdot \overrightarrow{m}) = f(\overrightarrow{m}), \quad \forall\, g \in G. \tag{4.7}$$

Let us suppose a smooth path in the parameter space $p(t) = \left(p_1(t), p_2(t), ..., p_{n_p}(t)\right)$. Thus, we have also:

$$g(t) = g\left(p_1(t), p_2(t), ..., p_{n_p}(t)\right). \tag{4.8}$$

By inserting expression (4.8) in equation (4.7) it is straightforward to show that:

$$\frac{d}{dt} f\left(\overrightarrow{m}'\left(t\right)\right) = 0, \tag{4.9}$$

otherwise f would not be an invariant. From the Chain Rule:

$$\sum_{i=1}^{n_m} \sum_{j=1}^{n_p} \frac{\partial f}{\partial m_i'} \frac{\partial m_i'}{\partial p_j} \frac{dp_j}{dt} = 0. \tag{4.10}$$

Two observations are fundamental at this point. Firstly, the differential equation (4.10) should be satisfied for any $g\left(t\right) = g\left(p_1\left(t\right), p_2\left(t\right), ..., p_{n_p}\left(t\right)\right)$. Thus for all $\frac{dp_j}{dt}$, it is necessary that:

$$\sum_{i=1}^{n_m} \frac{\partial f}{\partial m_i'} \frac{\partial m_i'}{\partial p_j} = 0, \text{ for all } j = 1, 2, ..., n_p. \tag{4.11}$$

The second observation is a Lie Group result. Simply stated, it is enough to solve the equation (4.11) for $t = 0$ [100]. Thus we finally have the following differential equations for f :

$$\sum_{i=1}^{n_m} \frac{\partial m_i'}{\partial p_j} \mid_{p=0} \left(\frac{\partial f}{\partial m_i}\right) = 0, j = 1, 2, ..., n_p. \tag{4.12}$$

The solutions of this system are the desired invariants. The differential operators:

$$\sum_{i=1}^{n_m} \frac{\partial m_i'}{\partial p_j} \mid_{p=0} \left(\frac{\partial}{\partial m_i}\right), \tag{4.13}$$

are called *infinitesimal generators* of the underlying group action [95, 100].

The expression (4.12) is an explicit representation of the Lie Derivative of a function f with respect to the infinitesimal generators given by expression (4.13). Precisely stated, equation (4.13) is a representation of the following fundamental theorem for invariance:

Theorem 1: Let G be a Lie Group of transformations. A real-valued function f is invariant under the group transformations (or action) if and only if the Lie derivative of f with respect to any infinitesimal generator, v, of the group G, vanishes, that is, $L_v\left(F\right) = 0$, where $L_v\left(.\right)$ denotes the Lie derivative with respect to a vector field v.

4.4 Statistical Learning Models

In this section we discuss some aspects os statistical learning theory related to the Support Vector Machine (**SVM**) and Perceptron, the basic unit for neural networks. The goal is to set a framework for the application of SVM and neural nets for the initialization of deformable models. The material to be presented follows the references [101, 113, 116].

4.4.1 Perceptron Model

The Perceptron is the first logical neuron. Its development starts with the work of W. S. McCulloch and W.A. Pitts in 1943 [116]. It describes the fundamentals functions and structures of a neural cell reporting that a neuron will fire an impulse only if a threshold value is exceeded.

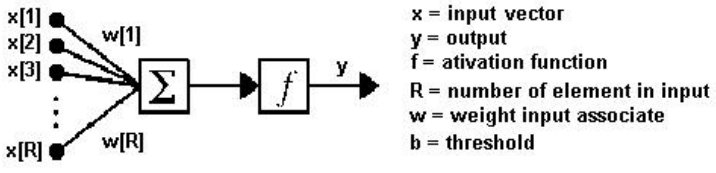


Figure 4.1: McCulloch-Pitts neuron model.

Figure 4.1 shows the basic elements of McCulloch-Pitts model: x is the input

vector, w are weights associated, y is output, R is number of elements in input and f is the activation function, named *decision function* in statistical learning theory, that determines the value in output. A simple choice for f is the signal function $sgn(.)$. In this case, if the sum, across all the inputs with its respective weights exceeds the threshold b the output is 1 else the value of y is -1, that is:

$$y = sgn(\sum_{i=1}^{R} w_i x_i - b). \tag{4.14}$$

But the McCulloch-Pitts neuron did not have a mechanisms for *learning*. Based on biological evidences, D.O. Hebb suggested a rule to adapt the weights input, which is interpreted as learning rule for the system [116]. This biological inspired procedure can be expressed in the following manner:

$$w_i^{new} = w_i^{old} + \Delta w_i; \quad \Delta w_i = \eta(y^{desired} - y)x_i, \tag{4.15}$$

where w^{new} and w^{old} are adapted weights and initials weights respectively, η is a real parameter to control the rate of learning and $y^{desired}$ is the desired (know) output. This *learning rule* plus the elements of Figure 4.1 is called the perceptron model for a neuron. It was proposed by F. Rosenblatt, at the end of of 1950s.

Then, the learning typically occurs through training, or exposure to a know set of input/output data. The training algorithm iteratively adjusts the connection weights $\{w_i\}$ analogous to synapses in biological nervous. These connection weights store the knowledge necessary to solve specific problems.

Geometrically, the connection weights $\{w_i\}$ and the the threshold b define a plane in a high dimensional space that separates the samples of distinct groups. Such *separating hyperplane* can be further used for classification on a new input vector x. Therefore, the learning process means to be able to adjust initial weights towards the separating hyperplane. Besides, it can be demonstrated that, if we can choose a small margin δ, such that:

$$\sum_{i=1}^{R} w_i x_i - b > \delta, \quad if \quad y = 1, \tag{4.16}$$

$$\sum_{i=1}^{R} w_i x_i - b < -\delta, \quad if \quad y = -1, \tag{4.17}$$

then, the number of times, T, that the rule defined by expression (4.15) is applied (number of iteractions) is bounded by:

$$T \leq \frac{1}{\delta^2}. \tag{4.18}$$

More precise bounds can be found in [101]. Perceptrons can be organized in a network, like in Figure 4.2. The generated neural net can be used as a classifier in general classification problems (multi-class and non-linear classifier).

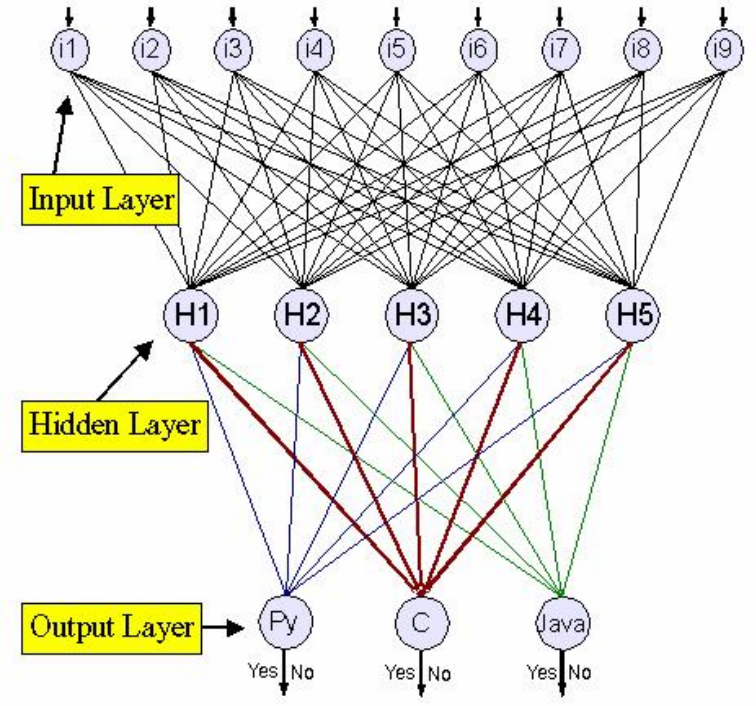


Figure 4.2: Neural Network with three layers.

4.4.2 Support Vector Machines

In section 4.4.1 it became clear the importance of separating hyperplanes methods for learning algorithms. In this section we will present a special type of separating hyperplanes with optimality properties. So, given a training set:

$$S = \{(y_1, x_1), ..., (y_m, x_m)\}, \quad x \in \Re^n, \quad y \in \{-1, 1\}, \tag{4.19}$$

we say that the subset I for which $y = 1$ and the subset II for which $y = -1$ are separable by the hyperplane:

$$x * \phi = c, \tag{4.20}$$

if there exists both a unit vector ϕ ($|\phi| = 1$) and a constant c such that the inequalities:

$$x_i * \phi > c, \quad x_i \in I, \tag{4.21}$$

$$x_j * \phi < c, \quad x_j \in II, \tag{4.22}$$

hold true ("$*$" denotes the usual inner product in $\Re^m$). Besides, let us define for any unit vector ϕ the two values:

$$c_1(\phi) = \min_{x_i \in I} (x_i * \phi), \tag{4.23}$$

$$c_2(\phi) = \max_{x_j \in II} (x_j * \phi). \tag{4.24}$$

The Figure 4.3 represents the dataset and the hyperplanes defined by ϕ and the values c_1, c_2 defined in expressions (4.23)-(4.24):

In this figure the points P_1 and P_2 gives the solutions of problems (4.23)-(4.24), respectively, and the planes π_1 and π_2 are defined by:

$$x * \phi = c_1, \tag{4.25}$$

$$x * \phi = c_2. \tag{4.26}$$

Now, let us consider the plane π, parallel to π_1,π_2, with the property:

$$d_\pi(P_1) = d_\pi(P_2), \tag{4.27}$$

where $d_\pi(P)$ means the Euclidean distance from a point P to a plane π. This plane is the hyperplane that separates the subsets with maximal marging. Expression (4.27) can be written as:

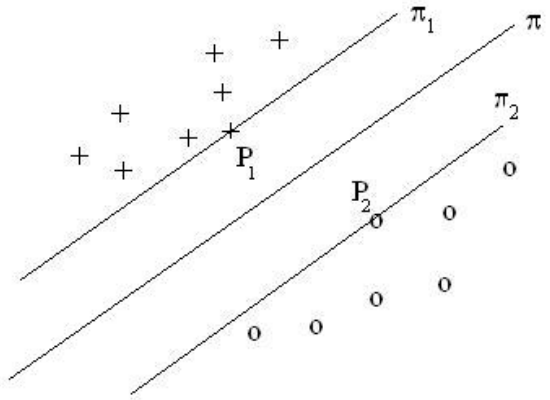


Figure 4.3: Separating hyperplane π and its offsets π_1, π_2.

$$\left|\frac{P_1 * \phi - c}{|\phi|}\right| = \left|\frac{P_2 * \phi - c}{|\phi|}\right|. \tag{4.28}$$

If we suppose $P_1 * \phi - c \geq 0$ then we have $P_2 * \phi - c \leq 0$. So, by remembering that $|\phi| = 1$, the expression (4.28) becomes:

$$(P_1 * \phi - c) + (P_2 * \phi - c) = 0,$$

then, by using expressions (4.25)-(4.26) we finally obtain:

$$c = \frac{c_1(\phi) + c_2(\phi)}{2}. \tag{4.29}$$

Besides, let us call the $d_{\pi_1}(\pi_2)$ the distance between the planes π_1 and π_2, which can be computed through the distance between the point P_1 and the plane π_2, given by:

$$d_{\pi_1}(\pi_2) \equiv d_{\pi_2}(P_1) = \frac{(P_1 * \phi - c_2)}{|\phi|}, \tag{4.30}$$

By using expression (4.25), this equation becomes:

$$d_{\pi_1}(\pi_2) = c_1 - c_2. \tag{4.31}$$

We call the *maximum marging hyperplane* or the *optimal hyperplane* the one, define by the unit vector ϕ_0 that maximizes the function:

$$\begin{aligned} \rho(\phi) &= \frac{c_1(\phi) - c_2(\phi)}{2}, & (4.32)\\ |\phi| &= 1. & (4.33) \end{aligned}$$

The corresponding separating plane π has a constant c given by equation (4.29).

Now let us consider another version of the optimization problem above. Let us consider a vector ψ such that $\psi/|\psi| = \phi$. So, equations (4.25)-(4.26) become:

$$x_i * \psi > |\psi| c_1, \quad x_i \in I, \tag{4.34}$$

$$x_j * \psi < |\psi| c_2, \quad x_j \in II \tag{4.35}$$

Let us suppose that there is a constant b_0 such that $|\psi| c_1 \geq 1 - b_0$ and $|\psi| c_2 \leq -1 - b_0$. Then, we can rewrite expressions (4.34)-(4.35) as:

$$x_i * \psi + b_0 \geq 1, \quad y_i = 1, \tag{4.36}$$

$$x_j * \psi + b_0 \leq -1, \quad y_j = -1. \tag{4.37}$$

To understand the meaning of b_0 it is just a matter of using the fact that the equality in (4.36) holds true for P_1 and the equality in (4.37) is true for P_2. Therefore, it is straightforward to show that:

$$b_0 = -|\psi| \left(\frac{c_1(\phi) + c_2(\phi)}{2} \right) = -|\psi| c. \tag{4.38}$$

So, by substituting this equation in expressions (4.36)-(4.37) one obtains:

$$x_i * \phi \geq c + \frac{1}{|\psi|}, \quad y_i = 1, \tag{4.39}$$

$$x_i * \phi \leq c - \frac{1}{|\psi|}, \quad y_j = -1. \tag{4.40}$$

These expressions mean that we suppose that we can relax the constant c through the value $(1/\left|\psi\right|)$ without loosing the separating property. But, the vector ψ is not an unit one. Therefore the distance (4.30) can be obtained by:

$$d_{\pi_1}(\pi_2) = \frac{(1-b_0)-(-b_0-1)}{|\psi|} = \frac{2}{|\psi|}. \tag{4.41}$$

In order to maximize this distance (and also maximize the function $\rho(\phi)$ in equation (4.32)) we must minimize the denominator in expression (4.41). So, we get an equivalent statement to define the optimal hyperplane: Find a vector ψ_0 and a constant (threshold) b_0 such that they satisfy the constraints:

$$x_i * \psi_0 + b_0 \geq 1, \quad y_i = 1, \tag{4.42}$$

$$x_j * \psi_0 + b_0 \leq -1, \quad y_j = -1. \tag{4.43}$$

and the vector ψ_0 has the smallest norm:

$$|\psi| = \psi * \psi. \tag{4.44}$$

We shall simplify the notation by rewriting the constraints (4.42)-(4.43) in the equivalent form:

$$y_i(x_i * \psi_0 + b_0) \geq 1, \quad i = 1, 2, ..., m. \tag{4.45}$$

In order to solve the quadratic optimization problem stated above, it is used in [101] the Kuhn-Tucker Theorem, which generalizes the Lagrange multipliers for convex optimization. The corresponding Lagrange function is:

$$L(\psi, b, \alpha) = \frac{1}{2}\psi * \psi - \sum_{i=1}^{m} \alpha_i \left(y_i\left((x_i * \psi_0) + b_0\right) - 1\right), \tag{4.46}$$

where α_i are the Lagrange multipliers. Following the usual theory, the minimum points of this functional must satisfy the conditions:

$$\frac{\partial L}{\partial \psi} = \psi - \sum_{i=1}^{m} y_i \alpha_i x_i = 0, \tag{4.47}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{m} y_i \alpha_i = 0. \tag{4.48}$$

If we substitute (4.47) into the functional (4.46) and take into account the result (4.48) we finally render the following objective function:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j (x_i * x_j). \tag{4.49}$$

We must maximize this expression in the nonnegative quadrant $\alpha_i \geq 0$, $i = 1, 2, ..., m$, under the constraint (4.48) . In [101] it is demonstrate that the desired solution is given by:

$$\psi_0 = \sum_{i=1}^{m} y_i \alpha_i x_i, \tag{4.50}$$

$$b_0 = \max_{|\phi|=1} \rho(\phi), \tag{4.51}$$

subject to:

$$\sum_{i=1}^{m} y_i \alpha_i = 0, \tag{4.52}$$

$$\alpha_i (y_i ((x_i * \psi_0) + b_0) - 1) = 0, \quad i = 1, 2, ..., m, \tag{4.53}$$

$$\alpha_i \geq 0. \tag{4.54}$$

The expression (4.53) states the Kuhn-Tucker conditions. By observing these conditions one concludes that the nonzero values of α_i, $i = 1, 2, ..., m$, correspond only to the vectors x_i that satisfy the equality:

$$y_i ((x_i * \psi_0) + b_0) = 1. \tag{4.55}$$

These vectors are the closest to the optimal hyperplane. They are called *support vectors*. The separating hyperplane can be written as:

$$f(x, \alpha_0) = \sum_{i=1}^{m} y_i \alpha_i^0 (x_i * x) + b_0, \tag{4.56}$$

where α_i^0, $i = 1, 2, ..., m$,satisfy the constraints (4.52)-(4.54). So, we can construct a decision function that is nonlinear in the input space:

$$f(x, \alpha) = sign\left(\sum_{i=1}^{m} y_i \alpha_i^0 (x_i * x) + b\right), \tag{4.57}$$

Now we describe two generalizations for the above approach..

Generalizing the SVMs

According to Vapnik [101], a *support vector machine* implement the following idea: "It maps the input vectors x into a high-dimensional *feature space* Z through some nonlinear mapping, chosen a priori. In the space Z an optimal separating hyperplane is constructed."

The key idea behind this proposal comes from the inner product "$*$" in equation (4.56). Firstly, if we map a vector $x \in \Re^n$ into a Hilbert space Z with coordinates $(z_1, z_2, ...)$ we get another representation for the feature space given by:

$$z_1(x), z_2(x), ..., z_n(x), ..., \tag{4.58}$$

Then, taking the usual inner product in the Hilbert space we get an equivalent representation for the inner product in the $\Re^n$:

$$z^1 * z^2 = \sum_{i=1}^{\infty} a_i z_i^1\left(x^1\right) z_i^2\left(x^2\right) \Longleftrightarrow K\left(x^1, x^2\right), \quad a_i \geq 0 \tag{4.59}$$

where $K(x^1, x^2)$ is a symmetric function satisfying the condition:

$$\int_C \int_C K(u, v)\, g(u)\, g(v)\, dudv \geq 0,$$

for all $g \in L^2(C)$, C being a compact subset of $\Re^n$. In this case we say that $K(u, v)$ is the kernel that generates the inner product for the feature space.

Therefore, we can generalize expression (4.57) by using the inner product defined by the kernel K :

$$f(x, \alpha) = sign\left(\sum_{supportvectors} y_i \alpha_i^0 K(x_i * x) + b\right), \tag{4.60}$$

or, equivalently, we can use the linear decision function in the feature space Z :

$$f(x,\alpha) = \\ sign\left[\sum_{supportvectors} y_i \alpha_i^0 \left(\sum_{r=1}^{\infty} a_r z_r\left(x^i\right) z_r(x)\right) + b\right], \tag{4.61}$$

These expressions define the SVM method [101, 117]. In summary, SVM seeks to find the hyperplane defined by equation (4.61) which separates positive and negative observations with the maximum margin.

Chapter 5

Energy Minimization Models

5.1 Introduction

In this section we focus mainly in discrete models that use an explicit energy minimization approach; that means, an energy functional is defined and the evolution equation is derived from this functional through a kind of steepest descent technique.

5.2 Invariant Snakes

When prior knowledge of shape is available, information concerning the shape of the desired object can be incorporated into the snake formulation. The so called Shaped-Based Active Contour Models are suitable to detect and to locate objects which have been distorted due to some kind of geometry transformations. They can be considered a special case of active shape models, which have been used for detecting structures in medical images (see [102] and references therein).

Basically, the shape information is incorporated in a *prototype* and a correspondence between the snake and the prototype must be established.

The Generalized Active Contour Model [103] is a known example of such an approach. In this case, a shape matrix is incorporated to the model to create a bias towards a particular type of contour. The corresponding internal energy is

invariant to scale transformations but it is not affine-invariant.

This limitation is addressed by the AI-Snake [19]. This is a shape-based snake model which incorporates an internal energy function that is defined in terms of local and global affine-invariant features. Let us see some details of this model.

Firstly, it is observed in [19] that the gaussian curvature is not affine invariant. To show this, let us take a parametric (smooth) curve $c(\tau) = (x(\tau), y(\tau))$. Hence, the gaussian curvature $k(\tau)$ is given by:

$$k(\tau) = \frac{\dot{x}(\tau)\,\ddot{y}(\tau) - \ddot{x}(\tau)\,\dot{y}(\tau)}{\left[\left(\dot{x}(\tau)\right)^2 + \left(\dot{y}(\tau)\right)^2\right]^{3/2}} \tag{5.1}$$

If s represents the arc length of c then:

$$\left(\frac{ds}{d\tau}\right)^2 = \left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2,$$

and the expression (5.1) can be rewritten as:

$$k(\tau) = \frac{\begin{vmatrix} \dot{x}(\tau) & \ddot{x}(\tau) \\ \dot{y}(\tau) & \ddot{y}(\tau) \end{vmatrix} \cdot (d\tau)^3}{(ds)^3} = \frac{ang(x,y)}{dist(x,y)} \tag{5.2}$$

where:

$$ang(x,y) = \begin{vmatrix} \dot{x}(\tau) & \ddot{x}(\tau) \\ \dot{y}(\tau) & \ddot{y}(\tau) \end{vmatrix} (d\tau)^3; \qquad dist(x,y) = (ds)^3. \tag{5.3}$$

Let us see the behaviors of ang and $dist$ under affine transformations given by the general form:

$$\begin{pmatrix} u(\tau) \\ v(\tau) \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x(\tau) \\ y(\tau) \\ 1 \end{pmatrix} \tag{5.4}$$

By inserting this expression in ang and $dist$ we can show that they are transformed as follows:

$$ang\left(u,v\right)=\left(a_{11}a_{22}-a_{12}a_{21}\right)\cdot ang\left(x,y\right). \tag{5.5}$$

$$dist\left(u,v\right)=\left(\left(du\right)^{2}+\left(dv\right)^{2}\right)^{3/2}=\left(\left(a_{11}dx+a_{12}dy\right)^{2}+\left(a_{21}dx+a_{22}dy\right)^{2}\right)^{3/2}. \tag{5.6}$$

Expression (5.5) shows that ang is affine invariant if appropriately normalized. However, $dist$ cannot be made affine invariant except when the scaling parameters in both directions are identical. Thus, the Gaussian curvature is not affine invariant.

A curvature-like affine invariant feature can be derived from a discrete version of $(1/2)\,ang$. To demonstrate this, let us consider the following finite difference approximations for the first and second derivatives of $x\left(\tau\right)$:

$$\dot{x}\left(\tau\right) \doteq \frac{x\left(\tau\right)-x\left(\tau-\Delta\tau\right)}{\Delta\tau}, \tag{5.7}$$

$$\ddot{x}\left(\tau\right) \doteq \frac{x\left(\tau\right)-2x\left(\tau-\Delta\tau\right)+x\left(\tau-2\Delta\tau\right)}{\left(\Delta\tau\right)^{2}}, \tag{5.8}$$

(the same for y). Inserting these expressions in $(1/2)\,ang\left(x,y\right)$ yields the following result:

$$\frac{1}{2}ang\left(x,y\right)\doteq\frac{1}{2}\begin{vmatrix} x\left(\tau_1\right) & x\left(\tau_2\right) & x\left(\tau_3\right) \\ y\left(\tau_1\right) & y\left(\tau_2\right) & y\left(\tau_3\right) \\ 1 & 1 & 1 \end{vmatrix}, \tag{5.9}$$

where $\tau_1=\tau-2\Delta\tau, \tau_2=\tau-\Delta\tau, \tau_3=\tau$. This expression gives the area of a triangle formed by the three vertices: $\left(x\left(\tau_1\right),y\left(\tau_1\right)\right),\left(x\left(\tau_2\right),y\left(\tau_2\right)\right),\left(x\left(\tau_3\right),y\left(\tau_3\right)\right)$. We denote this quantity by $area_{\tau_1\tau_2\tau_3}$ (collinear cases are particularly considered in [19]). We can demonstrate that it is invariant under affine transformations. Let us denote $area^{transf}_{\tau_1\tau_2\tau_3}$ the affine transformation of this quantity:

$$area^{transf}_{\tau_1\tau_2\tau_3}=\frac{1}{2}\begin{vmatrix} u\left(\tau_1\right) & u\left(\tau_2\right) & u\left(\tau_3\right) \\ v\left(\tau_1\right) & v\left(\tau_2\right) & v\left(\tau_3\right) \\ 1 & 1 & 1 \end{vmatrix}. \tag{5.10}$$

By inserting transformations given by (5.4) in this expression, we can obtain:

$$area^{transf}_{\tau_1\tau_2\tau_3} = (a_{11}a_{22} - a_{12}a_{21})\, area_{\tau_1\tau_2\tau_3}. \tag{5.11}$$

Thus, the area of a triangle can be made affine-invariant if appropriately normalized. Starting from this fact, it was proposed in [19] the following affine invariant internal energy:

$$E_{suav} = \sum_{i=0}^{N-1} \alpha \frac{|area_i|}{AREA_{now}} \frac{AREA_{model}}{\frac{1}{2}d^2}, \tag{5.12}$$

where α is a parameter, $AREA_{now}$ is the area enclosed by the snake:

$$AREA_{now} = \frac{1}{2}\sum_{i=0}^{N-1} \begin{vmatrix} u_i & u_{i+1} & u_{cent} \\ v_i & v_{i+1} & v_{cent} \\ 1 & 1 & 1 \end{vmatrix}; \qquad \begin{pmatrix} u_{cent} \\ v_{cent} \end{pmatrix} = \frac{1}{N}\sum_{i=0}^{N-1} \begin{pmatrix} u_i \\ v_i \end{pmatrix}.$$

with $\{(u_0, v_0), (u_1, v_1), ..., (u_{i-1}, v_{i-1}), (u_i, v_i), (u_{i+1}, v_{i+1}), ..., (u_{N-1}, v_{N-1})\}$ being snake points. $AREA_{model}$ is the area of the prototype, d is the mean distance between snaxels and:

$$area_i = \frac{1}{2} \begin{vmatrix} u_{i-1} & u_i & u_{i+1} \\ v_{i-1} & v_i & v_{i+1} \\ 1 & 1 & 1 \end{vmatrix} \tag{5.13}$$

5.2.1 Reparameterization and Invariance

First of all, we must observe that expression for ang in (5.3) is not invariant under reparameterization. In fact, if $\tau = \phi(\varsigma)$ then the derivatives of x of order zero, one and two are transformed according to the matrix (and analogously for y):

$$\begin{pmatrix} x(\varsigma) \\ \dot{x}(\varsigma) \\ \ddot{x}(\varsigma) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & a & 0 \\ 0 & b & a^2 \end{pmatrix} \begin{pmatrix} x(\tau) \\ \dot{x}(\tau) \\ \ddot{x}(\tau) \end{pmatrix} \tag{5.14}$$

where $a = \frac{d\phi}{d\varsigma}$ and $b = \frac{d^2\phi}{d\varsigma^2}$. If ϕ is monotone function then this matrix represents a Lie Group of dimension $(2+1)$ called *reparameterization group* (in a general development, the dimension would be $(m+1)$ where m is the highest-order derivative considered). Under the group action, the quantity ang is transformed to ang^{transf} given by:

$$ang^{transf} = a^3 ang, \tag{5.15}$$

Thus, it is not an invariant (it is called a *relative invariant* because the factor a^3 depends upon the group parameters only).

Reparameterization for snakes has been deeply discussed in the literature [15, 95]. In general, snake models are not invariant under the reparameterization group. Thus, the internal forces derived from Euler-Lagrange equations change the form due to the group action (more formally, they are not covariant), which is an unwanted effect.

For affine-invariant snakes, a complete description of the reparameterization problem comes from the Lie Theory. In this case, we have to account for two (Lie) groups: the reparameterization and affine transformations. Is there a common invariant for both groups?

To the best of our knowledge, the answer is negative. The intersection set is not null only for the subset of the unimodular transformations ($a_{11}a_{22} - a_{12}a_{21} = 1$). See [95] for more details.

Despite of this, a couple of interesting facts appear when discussing invariant snakes in the context of integral invariants, a special application of the Lie Groups theory summarized on section 4.3.

Integral invariants are globally defined while differential invariants (depend only on derivatives of the coordinate functions) are local features. Integral invariants are suitable for matching curves. However, they are limited if occlusions appear. This limitation has motivated the utilization of semi-local integral invariants [104].

Thus, let us consider the integral:

$$I = \int_{w_1}^{w_2} F d\tau \ , \tag{5.16}$$

where F is a function along the curve $c(\tau)$. A sufficient condition for I to be invariant under a transformation group can be stated as follows:

Proposition 1: The integral of a function F, with respect to a parameter τ, is invariant under a transformation group if the Lie derivative of I and τ, with respect to any infinitesimal generator v of the group, vanishes: $L_v(F) = 0, \quad L_v(\tau) = 0$ (see Appendix A).

That means that both F and τ are invariant under the specified transformation group.

From equation (5.5), section 2.1, it follows that the expression:

$$|c_\tau \quad c_{\tau\tau}| = \begin{vmatrix} \dot{x}(\tau) & \ddot{x}(\tau) \\ \dot{y}(\tau) & \ddot{y}(\tau) \end{vmatrix}. \tag{5.17}$$

is a relative invariant under general affine transformations. Another relative invariant under the action of this group is the affine arc-length μ of a curve c, defined as follows:

$$|c_\mu \quad c_{\mu\mu}| = 1 \Rightarrow d\mu = |c_\tau \quad c_{\tau\tau}|^{1/3} d\tau, \tag{5.18}$$

where, $c_\mu, c_{\mu\mu}$ denote the first and second derivatives of c with respect to μ and $|c_\mu \quad c_{\mu\mu}|$ is a determinant, similar to expression (5.17) (the right expression in (5.18) is obtained by applying the chain rule to the left one [104]).

Thus, if $\tilde{\mu}$ is the transformed quantity then, from (5.18) we find that:

$$\int d\tilde{\mu} \quad = \quad |A|^{1/3} \int d\mu, \tag{5.19}$$

where $|A|$ is the determinant of the affine transformation defined in expression (5.4). So, from Property 1, it is a relative integral invariant under affine transformations and can be made affine invariant through a convenient normalization factor.

Besides, we shall observe that it follows from equation (5.18) that:

$$d\mu^3 = |c_\tau \quad c_{\tau\tau}| \, d\tau^3 = ang(x, y), \tag{5.20}$$

according to expressions (5.2)-(5.3).

Moreover, another interesting fact appears if we rewrite the $area_i$ (expression (5.13)) in the form:

$$area_i = \frac{1}{2} |c(\mu_i + \Delta\mu) - c(\mu_i) \quad c(\mu_i - \Delta\mu) - c(\mu_i)|, \tag{5.21}$$

where we are supposing a reparameterization of the curve: $\tau = \tau(\mu)$.

We shall observe that this expression can be obtained by the following integral:

$$area_i = \frac{1}{2}\int_{\mu_i-\Delta\mu}^{\mu_i+\Delta\mu} |c_\mu(\mu) \quad c(\mu_i+\Delta\mu) - c(\mu_i)|\, d\mu\,, \tag{5.22}$$

which is also an integral similar to expression (5.16) as well as an affine semi-local integral invariant, as shown in [104]. It was used in this reference for matching curves. Thus we can postulate its utilization to match the actual snake with the prototype, instead of the local features used in the AI-Snake model [19]. Besides, explorations over expressions (5.19)-(5.22) should be done to find out the potential of integral invariants for snake models. These are further directions for this work.

5.3 Dynamic Programming and Greedy Snakes

Dynamic programming (**DP**) has been used in snake models to address the sensitivity to local minima [56]. DP ensures global optimality of the solution and do not require estimates of higher order derivatives which improves the numerical stability. In particular, the Viterbi algorithm has been applied for dual approaches [33, 42, 38]. In this algorithm the search space is constructed by discretizing each curve in N points and establishing a matching between them. Each pair of points is then connected by a segment which is subdivided in M points. This process provides a discrete search space, with NM points, that is pictured in the Figure 5.1:

The target boundary is then determined by minimizing the following energy functional [33]:

$$E_{snake} = \sum_{i=0}^{N-1} E_i; \tag{5.23}$$

$$E_i = \alpha E_{int}(v_{i-1}, v_i, v_{i+1}) + \beta E_{ext}(v_i) + \lambda E_{line}(v_i)\,, \tag{5.24}$$

with E_{int}, E_{ext} and E_{line} defined as follows:

$$E_{int}(v_{i-1}, v_i, v_{i+1}) = \left(\frac{v_{i+1} - 2v_i + v_{i-1}}{\|v_{i+1} - v_{i-1}\|}\right)^2 = \left(\frac{2V_1}{\|v_{i+1} - v_{i-1}\|}\right)^2, \tag{5.25}$$

$$E_{ext}(v_i) = -\|\nabla I(v_i)\|^2; \quad E_{line}(v_i) = \pm I(v_i)\,, \tag{5.26}$$

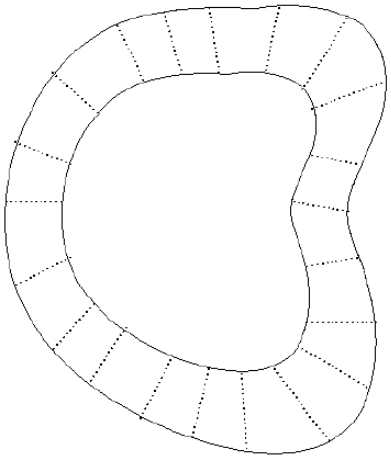


Figure 5.1: Search space obtained through a matching between inner and outer snakes.

where V_1 is defined on Figure 3.1.

The real parameters α, β and λ must be chosen in advance in order to weigh the influence of each term in expression (5.24). The model energy, given by expression (5.24), can be derived from expression (3.24) by setting $w_1 = 0$ and adding E_{line} to the external energy. Therefore, the solution may be attracted to points with high gradient, due to E_{ext}, as well as to be attracted to light (signal $-$) or dark lines (signal $+$) due to the term E_{line}. The energy E_{int} is a smoothing term, based on equation (3.22), but now taking $\Delta s_i = \sqrt{\|v_{i+1} - v_{i-1}\|}$ and $h_i = 1$ in expression (3.24). The advantage of these choices is that the functional E_{int} becomes invariant to scale transformations (remains unchanged if we replace v_i by $q.v_i$) which avoids the unwanted contraction effect observed in the original snake model (section 3.2).

When minimizing expression (5.23) we want the solution to be smooth and attracted to the desired feature (light/dark lines or strong edges). The corresponding recurrence relation is the classical one [16]:

$$S_i\left(v_{i+1}, v_i\right) = \min_{v_{i-1}} \left\{S_{i-1}\left(v_i, v_{i-1}\right) + E_{i-1}\right\}. \tag{5.27}$$

where $S_i\left(v_{i+1}, v_i\right)$ is the energy of the optimum path connecting v_i to v_0.

This idea can be adapted for evolutionary approaches through the following steps: (1) Define a neighborhood around each snaxel and find the minimal energy contour for the set of pixels considered around each point. (2) If the contour energy has not changed from previous iteration, exit. (3) Move points to newly computed locations. (4) Go to 1 (see [16] for details).

If the two snakes in Figure 5.1 are close to each other, there is another search based method that can be efficient and less expensive then the Viterbi one. Firstly, we compute a curve located in-between the two fronts by taking the midpoint of each segment of the search space in Figure 5.1. The obtained curve can be used to initialize a snake model based on a greedy algorithm that works as follows. For each snaxel v_i we take a 3×3 neighborhood V, and for each pixel $p \in V$ we compute:

$$E(p) = \alpha E_{int}(v_{i-1}, p, v_{i+1}) + \beta E_{ext}(p) + \lambda E_{line}(p). \tag{5.28}$$

Then, we solve the problem:

$$p_o = \arg\min \{E(p); \quad p \in V\}. \tag{5.29}$$

If $E(p_o) < E(v_i)$ then $v_i \leftarrow p$. The algorithm can be summarized as:

Algorithm 1 Greedy Snake Procedure

for all snaxel v_i **do**

take a 3×3 neighborhood V

$p_o = \arg\min \{E(p); \quad p \in V\}$;

if $E(p_o) < E(v_i)$ **then**

$v_i \leftarrow p_o$.

The snake position is updated following this procedure for all snaxels. A termination condition is achieved when snaxels do not move anymore (equilibrium position). Greed algorithms have been used in the context of snake models in order to improve computational efficiency [57, 58].

5.4 Original Dual Model

The dual snake methodology was firstly proposed in [17]. To obtain the conventional continuity and smoothness constraints, but removes the unwanted contraction force, a scale invariant internal energy function (shape model) is developed. In [17] a snake is considered as a particle system $v_i = (x_i, y_i), i = 0, ..., N-1$ whose particles are linked by internal constraints. The shape model is accomplished by the following internal energy:

$$E_R = \frac{1}{N}\sum_{i=1}^{N-1} E_{int}(v_i); \quad E_{int}(v_i) = \frac{1}{2}\left(\frac{\|e_i\|}{h}\right)^2, \tag{5.30}$$

where:

$$e_i = \frac{1}{2}(v_{i-1} + v_{i+1}) - v_i + \frac{1}{2}\theta_i R(v_{i-1} - v_{i+1}), \tag{5.31}$$

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

$$\theta_i = \cot\left(\frac{\varphi_i}{2}\right). \tag{5.32}$$

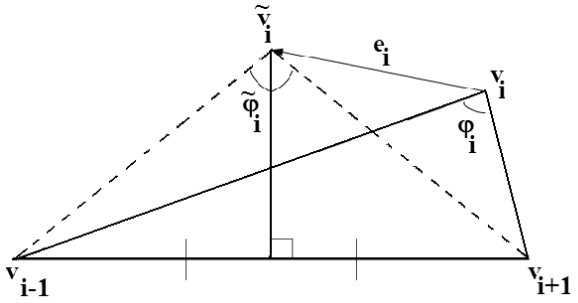


Figure 5.2: Geometric elements of the local shape model.

The Figure 5.2 helps to understand the geometric meaning of these elements. In this figure, the vector e_i is such that the triangle with vertices v_{i+1},$v_i + e_i$,v_{i-1} is isosceles. It is clear that E_R has a global minimum when $e_i = 0$, $i = 0, 1, ..., N-1$. From (5.31)-(5.32) it can be shown that this happens when:

$$\varphi_i = \pi\left(N-2\right)/2N, i = 0, 1, ..., N-1, \tag{5.33}$$

which are the internal angles of a regular polygon with vertices given by the points v_i [17]. The energy (5.30) can be also shown to be rotation, translation and scale invariant [17]. Therefore there is no tendency to contraction as already pointed out for the original snake model (section 3). With the internal energy given by expression (5.30), the curve is biased towards a regular polygon [17]. That is the main difference when comparing expression (5.30) with the internal energy of the original dual model, equation (5.25).

As before, the external energy is defined by:

$$E_{ext}\left(v_i\right) = -\left\|\nabla I\left(v_i\right)\right\|^2, \tag{5.34}$$

The total energy of the model is given by:

$$E = \frac{1}{N}\sum_{i=1}^{N-1}\left(\lambda E_{int}\left(v_i\right) + \left(1-\lambda\right)E_{ext}\left(v_i\right)\right), \tag{5.35}$$

where λ is a smoothing parameter which lies between 0 and 1 [17]. This expression guarantees that during the optimization process, the snake will seek for strong object boundaries, due to E_{ext}, whose shape resembles a regular polygonum, due to shape model given by expression (5.30). This fact makes easier to establish the correspondence (matching) between the points of the two contours because the form of the snake during the evolution is limited by the energy (5.30). The methodology takes advantage of this correspondence by proposing the driving force:

$$F_{driving} = g\left(t\right)\frac{u_i - v_i^t}{\left\|u_i - v_i^t\right\|}, \tag{5.36}$$

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 [17] is the following one, based on low velocity criterium stated in expression (3.27):

$$\max_i \left\|v_i^{t+1} - v_i^t\right\| < \delta, \tag{5.37}$$

where δ is a termination parameter. This expression is a discrete version of equation (7.16) if we take a first-order finite difference representation for the velocity $(\partial c/\partial t)$.

The dual approach consists in making the inner and outer contours evolve according the following algorithm: The contour with the highest energy is selected. If its motion remains below some termination condition then the driving force (5.36) 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.

The shape model given by expression (5.30) limits the application of the method for general shapes and topologies. Besides, the utilization of a match between the two contours puts restrictions to extend the technique for $3D$. These limitations are addressed by the Dual-T-Snakes [22, 42] approach, developed by some of us, which is described in section 6.3.

Other possibilities for the dual energy can be used. Following the reference [60] and the presentation of section 3 let us write the Dual energy in the form:

$$E(c) = w_1 E_{ten}(c(s)) + w_2 E_{rig}(c(s)) + \gamma E_{ext}(c(s)) + w_3 E_3, \qquad (5.38)$$

where E_{ten}, E_{rig} and E_{ext} are defined by expressions (3.3)-(3.5). The term E_3 is the model energy that is used to incorporate geometric constraints during the minimization process and can be given in the general form:

$$E_3 = \int_\Omega (c(s) - model(s))^2 \, ds, \qquad (5.39)$$

The parameter w_3 controls the weight of this term in expression (5.38) and $model(s)$ is the parametric shape information about the target. The key idea in this formulation is that the snake evolves seeking for edge points, due to the external energy E_{ext}, but the solution is biased towards the target model defined by $model(s)$.

5.5 Dual-Snake Model of High Penetrability

This model was proposed in [35] and is inspired in the model presented in [62] in which only the edge points are under consideration for the snaxel movements. The proposed dual snake model is based on three new schemes. The first one is to reinforce the conventional image force field by weighting each edge strength with the size of its connected component, that is, a cluster of edge points in which every two edge points can be connected through a path of edge points in the same cluster. The second scheme is a new external force, called discrete gradient flow (DGF). The third scheme is based on the idea of taking into account how close the snaxels are to the edge points to determine the energy state of a snake.

So, let Γ_1 and Γ_2 denote the inner and outer snakes, respectively (we drop the superscript "t" to simplify notation). In this model, both snakes have the same number of snaxels. Like in the model of section (5.7), the ith inner snaxel is denoted by $v_{1,i}$ while the ith outer snaxel is denoted by $v_{2,i}$. The energy of the inner and outer snakes are:

$$E_{snake}\left(\Gamma_l\right) =$$

$$\sum_i \left[\alpha_l E_{len}\left(v_{l,i}\right) + \beta_l E_{curvature}\left(v_{l,i}\right)\right] \quad +$$

$$\sum_i \left[\gamma_l S_l\left(v_{l,i}\right) E_{DGF}\left(v_{l,i}\right) + \delta_l E_{DS-Potential}\left(v_{l,i}\right)\right], \quad l = 1, 2, \tag{5.40}$$

where $\alpha_l, \beta_l, \gamma_l, \delta_l$ parameters to be set in advance. The internal energy E_{len} is defined like in expression (5.59) while the $E_{curvature}\left(v_{l,i}\right)$ is a finer scale version of expression (5.60), defined by:

$$E_{curvature}\left(v_{l,i}\right) = \cos^{-1}\left(\frac{\mathbf{u}_{l,i} \cdot \mathbf{u}_{l,i+1}}{\left|\mathbf{u}_{l,i}\right| . \left|\mathbf{u}_{l,i+1}\right|}\right), \tag{5.41}$$

where $\mathbf{u}_{l,i} = v_{l,i} - v_{l,i-1}$. The other terms in expression (5.40) are external energies to be defined next. The E_{DGF} term incorporates a definition of a new external field, called the discrete gradient flow (DGF), which aims to improve the convergence of the method.

The DGF may be derived from any conventional edge strength map (Sobel, Canny, etc.). For instance, for ultrasound images, a good choice is the modified trimmed mean (MTM) filter due to its simplicity and reasonably good denoising capability [63]. Besides, the edge strength map produced by conventional approaches can be augmented through morphological operations and a weighting scheme based on the concept of connected components (see [35] for details).

To construct the DGF map, the peaks of the obtained edge strength map are first determined, which define the candidate edge points. Then, the method takes the mean of the edge strengths within the region of interest and excludes the candidates with the edge strengths smaller than that mean.

Each pair of $v_{1,i}$ and $v_{2,i}$ has been designed to move toward each other along the straight line, called the **moving path** L_i, connecting both snaxels until they meet together (see Figure 5.3). In this model, only the edge points are under consideration for the snaxels to stop at.

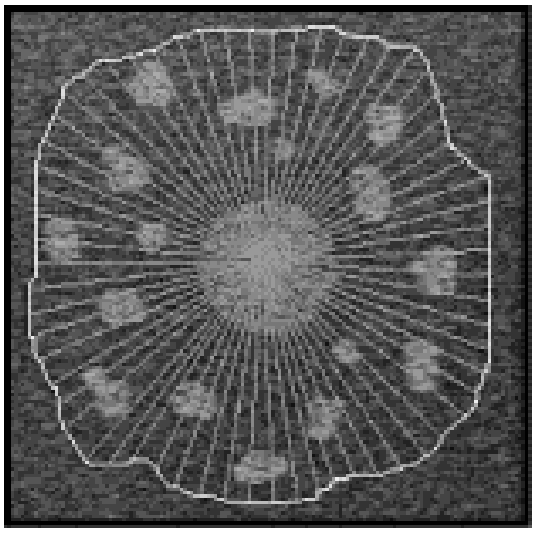


Figure 5.3: Moving paths connecting the pairs of snaxels $v_1(s_i)$ and $v_2(s_i)$ in the initial inner and outer snakes.

Let p_j^i denote the jth point in L_i and suppose that L_i has m_i points. Suppose also that L_i passes through the set of edge points $\Omega_i = \left\{p_{k_1}^i, p_{k_2}^i, \cdots, p_{k_{n_i}}^i\right\}$, where $1 \leq k_1 \leq k_2 \leq \cdots \leq k_{n_i} \leq m_i$, and $e\left(p_j^i\right)$ represents the edge strength of the point p_j^i. Then, for each $p_j^i \in \Omega_i \cup \left\{p_1^i, p_{m_i}^i\right\}$ recursively eliminate p_j^i if

$e\left(p_j^i\right) < e\left(p_{jl}^i\right)$ and $e\left(p_j^i\right) < e\left(p_{jr}^i\right)$, where $p_{jl}^i, p_{jr}^i \in \Omega_i \cup \left\{p_1^i, p_{m_i}^i\right\}$ and p_{jl}^i and p_{jr}^i are the nearest neighbors to the left and to the right of p_j^i. If Λ_i is the set of remaining points, then the DGF for all points in L_i is defined as:

$$DGF_i\left(p_j^i\right) = \frac{d\left(p_{jl}^i, p_j^i\right)}{d\left(p_{jl}^i, p_{jr}^i\right)} e\left(p_{jr}^i\right) + \frac{d\left(p_j^i, p_{jr}^i\right)}{d\left(p_{jl}^i, p_{jr}^i\right)} e\left(p_{jl}^i\right), \tag{5.42}$$

where $p_{jl}^i, p_{jr}^i \in \Lambda_i$ are defined as before and $d\left(\cdot,\cdot\right)$ means the Euclidean distance in L_i. Then, the term $E_{DGF}\left(v_{l,i}\right)$ in expression (5.40) at the point p_j^i in L_i is defined as $DGF_i\left(p_j^i\right)$.

The snake deformation in the dual-snake model proposed in [35] has been decomposed into three stages. In the first stage, the two snakes move toward each other independently until each snake reaches a local minimum based on the internal energies, E_{cont}, $E_{curvature}$ and E_{DGF}. Therefore, for all snaxels in this stage, the stability index S_1 is set to 1, δ is set to 0 and γ, α and β should be properly defined.

After both snakes come across the local minima, in the second stage, the snaxels most like to lie far from the boundary should be detected. In [35] the authors named **stability,** the measure that indicates, for each pair of inner and the outer snaxels, $v_{1,i}$ and $v_{2,i}$, each one will be encouraged to move forward to the other one by adding another energy term, (i.e., the DS-potential energy) denoted as $E_{DS-potential}$. The method is implemented through a **stability index** based on the concept that the closer a snaxel is to an edge, the more stable it would be. Therefore, for the kth snaxel $v_{l,k}$, in the snake Γ_l, $l = 1$ or $l = 2$, the method firstly computes:

$$\begin{aligned} D_{-1}\left(v_{l,k}\right) &= 1 \quad if \quad d_e\left(v_{l,k}\right) \leq 2, \\ D_{-1}\left(v_{l,k}\right) &= \frac{1}{d_e\left(v_{l,k}\right) - 1} \quad if \quad d_e\left(v_{l,k}\right) > 2, \end{aligned} \tag{5.43}$$

where $d_e\left(v_l\left(k\right)\right)$ is the distance from $v_l\left(k\right)$ to the nearest candidate edge point. Then, the stability index of the snake element $v_l\left(k\right)$ is defined as:

$$S_I\left(v_{l,k}\right) = \frac{1}{\left(2n_s + 1\right)} \sum_{j=-n_s}^{n_s} D_{-1}\left(v_{l,k}\right), \tag{5.44}$$

where $n_s$defines the radius of the neighborhood. Based on the stability index, for each snaxel $v_l(s_i)$, $l = 1$ or $l = 2$, the stability of $v_l(s_i)$, denoted as $S(v_l(s_i))$, is defined as:

$$S(v_{l,i}) = \overline{S}_l S_l(v_{l,i}) \left(\beta E_{curvature}(v_{l,i}) - \gamma w_{DGF}(v_{l,i}) E_{DGF}(v_{l,i}) \overline{E}_{DGF-1}(v_{l,i})\right), \tag{5.45}$$

where γ is another parameter and:

$$w_{DGF}(v_{l,i}) = 1 - \frac{1}{1 + \sqrt{d(v_{1,i}, v_{2,i})}}, \tag{5.46}$$

$$\overline{E}_{DGF-1}(v_{l,i}) = \frac{1}{N_L} \sum_{i=1}^{N_L} E_{DGF}(v_{l,i}), \tag{5.47}$$

with $\overline{S}_l$ been the mean of the stability indices of all snake elements in the snake Γ_l, and N_L is the number of moving paths. For each pair of $v_{1,i}$ and $v_{2,i}$, one snaxel, say $v_{1,i}$, is considered to be less stable than the other, say $v_{2,i}$, if $S(v_{1,i}) < S(v_{2,i})$ and:

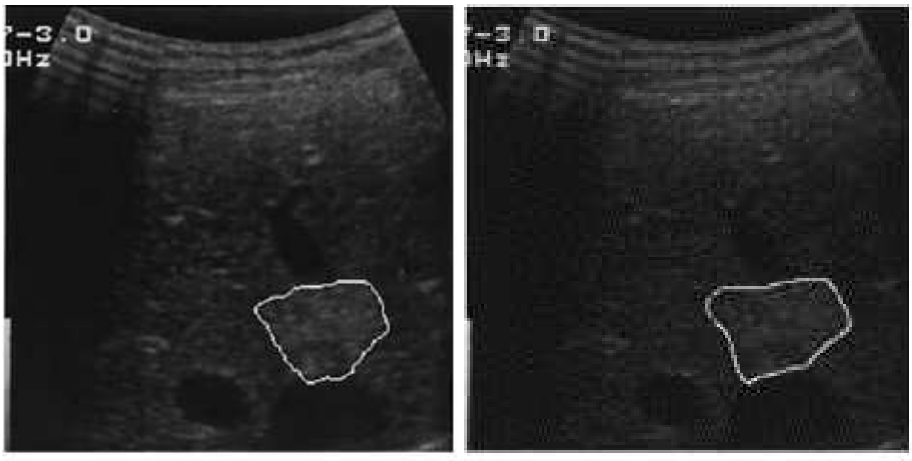
$$\frac{S(v_{2,i}) - S(v_{1,i})}{S(v_{2,i}) + S(v_{1,i})} < t_s, \tag{5.48}$$

where t_s is a predefined threshold. In this case, $v_{1,i}$ will try to move toward $v_{2,i}$ while $v_{2,i}$ remains at its current position. However, $v_{1,i}$ is allowed to move only if $E_{snake}(\Gamma_1)$, as defined in equation 5.40, decreases. The DS-potential energy is defined as:

$$E_{DS-potential}(v_{l,i}) = \frac{d(v_{1,i}, v_{2,i})}{\max_j d\left(p_1^j, p_{m_j}^j\right)}. \tag{5.49}$$

The role algorithm encompasses the following steps: (1) Define the inner and outer contours; (2) Construct the moving paths; (3) Construct an edge map and generate the corresponding DGF; (4) Both snakes evolve independently until reach the local minima ($S_I = 1$ and $\delta = 0$); (5) Compute the stability for each snaxel (expression (5.45)); (6) If one snaxel is less stable than the other one and the condition given in equation (5.48) is satisfied, move the less-stable snaxel toward the other one.

Figure 5.4.a shows the experimental result obtained with this method applied to a testing image containing a tumor. The Figure 5.4.b shows the same input image with the boundary drawn by a medical doctor. A visual comparison indicates a good agreement between the manual and automatic results. Qualitative comparisons can be also found in [35].



(a) (b)

Figure 5.4: (a)Boundary of a tumor extracted with the Dual-DGF. (b) The same image with a boundary drawn by a medical doctor.

5.6 Twin Models

The twin snakes concept is an extension to the original dual method. Twin snakes are designed for detecting two parallel contours simultaneously and are useful for line extraction in high resolution imagery. To achieve this goal an additional energy term is added to the energy functional in order to attract the snakes toward the target. They can be formulated through implicit or parametric models. In the case of the parametric twin model presented in [70], the energy functional has the form given by expressions (5.23)-(5.24), with:

$$E_{snake} = \sum_{i=0}^{N-1} E_i; \tag{5.50}$$

$$E_i = \alpha_i E_{el}\left(v_{i-1}, v_i, v_{i+1}\right) + \beta_i E_{cv}\left(v_{i-1}, v_i, v_{i+1}\right) + E_{img}\left(v_i\right) + E_d\left(v_i\right), \tag{5.51}$$

$$E_{el}\left(v_{i-1}, v_i, v_{i+1}\right) = \left(\left|v_{i-1} - v_i\right| - d\right)^2 + \left(\left|v_i - v_{i+1}\right| - d\right)^2, \tag{5.52}$$

$$E_{cv}\left(v_{i-1}, v_i, v_{i+1}\right) = \beta_i \left(v_{i-1} - 2v_i + v_{i+1}\right)^2, \tag{5.53}$$

where $\alpha_i \beta_i$ are parameters that controls the tension and regidity, respectively, of the model. The definition for E_{el} is equivalent to equation (5.59) for $d = 0$ while the energy E_{cv} is similar to expression (5.24) but without using normalization. The term E_d is new energy defined as:

$$E_d\left(v_i\right) = \delta\left(d_i - d_0\right)^2, \tag{5.54}$$

where δ is a weight factor, d_i is the actual distance to the twin partner at point i and d_0 the desired distance (constraint). The term $E_{img}\left(v_i\right)$ in expression (5.51) may be given by anyone of the expressions in (5.26). The energy $E_d\left(v_i\right)$ decreases when the snakes get closer to the target. Therefore, the solution is biased to extract a ribbon with width d_0. If we set $d_0 = 0$ then the energy E_{snake} becomes optimal when the two snakes have reached the same position, that means, they become identical. So, twin snakes approach can be adapted for single edge extraction. That is why the twin snakes model can be seen as generalization of the dual snakes.

For efficient calculation of distance d_i a snake has a pointer to the nearest vertex of the twin partner. This can be done by generating a Delaunay triangulation for which the lines of the snake polygon have to be edges of the triangles and each triangle has at most two vertices of the same snake. If the complete triangulation is known, the nearest neighbor to a vertex can be efficiently found by searching the neighboring nodes of the triangulation.

The energy minimization is based on the iterative dynamic programming algorithm [16] summarized on section 5.3. To reject local minima the twin method

uses the same method of the Dual-T-Snakes: compare snake positions and the snake energy, given by expression (5.50), normalized by the number of snaxels.

Twin as well as dual approaches can be extended in order to couple several contours and treat them as one deformable model [73]. Therefore, a set of n classical active contours $C = \{c_1, c_1, ..., c_n; \quad c_i : \Omega \rightarrow \Re^2\}$ is considered as an entity that is called coupled snake. Furthermore it is introduced in [73] a new internal energy E_{coup} to account for the coupling between the contours within C. It is defined as follows:

$$E_{coup} = \int_{\Omega} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \xi_{ij} dcoup\left(c_i\left(s\right), c_j\left(s\right)\right) \right) ds, \tag{5.55}$$

where ξ_{ij} is a weighting factor and the function $dcoup\left(c_i\left(s\right), c_j\left(s\right)\right)$ computes the coupling between the snakes $c_i\left(s\right)$ and $c_j\left(s\right)$ at s. To reduce the computational complexity, it is supposed that each classical contour in C has the same number of control-points and that each control-point of contour i is only interlinked to control-points of the other contours with the same index. For an unsymmetrical example, in which the contour c_1 pushes away c_2 while later has no effect on the former, we may define:

$$d_{coup}\left(c_i\left(s\right), c_j\left(s\right)\right) = 0, \quad if \quad i = 1, j = 2, \tag{5.56}$$

$$d_{coup}\left(c_i\left(s\right), c_j\left(s\right)\right) = \frac{1}{\max_s d\left(c_i\left(s\right), c_j\left(s\right)\right)}, \quad if \quad i = 1, j = 2, \tag{5.57}$$

where $d\left(c_i\left(s\right), c_j\left(s\right)\right)$ means the Euclidean distance between c_i and c_j at s. In [73] it is also proposed an extension of this model to segment multi-valued images.

5.7 Cell-Based Dual Snake Model

In [36] it is proposed a cell-based dual snake model for ultrasound image. Boundary extraction and segmentation for this kind of image are a much harder problem than for other image modalities, due to the speckle, the tissue-related textures, and the artifacts resulting from the ultrasonic imaging process. To address such difficulties it is proposed in [36] a model which is devised with three main stages, namely, cell generation, cell-based deformation and contour smoothing. In the

cell-generation stage, the immersion watershed algorithm [43] is used to generate the nonoverlapped cells. To alleviate the interference of speckle in cell generation, the speckle is reduced by using the multiscale Gaussian filters before computing the gradient map. The cell boundaries are defined as the watersheds of the dales formed in the gradient map of the speckle-reduced ultrasound images.

Once the cell decomposition of the image domain is performed, we must define the dual snakes and the evolution model. Thus, given an initial contour enclosing the region of interest (ROI), a set of minimum covering cells, $C_1, C_2, .., C_{N,}$ containing the ROI is found, based on the cells generated in the previous stage. Let Γ_1^0 and Γ_2^0 be the initial inner and outer snakes, respectively, which are pictured on Figure 5.5. The outer snake is defined as the outermost boundary of $\cup_{i=1}^{N} C_i$. Suppose that the center of the ROI is the cell C_1. Then, its boundary is the initial inner snake Γ_1^0, also represented on Figure 5.5.

Let Γ_1^t and Γ_2^t be the inner and outer snakes, respectively, at the deformation step t. Then, the model energy $E\left(\Gamma_1^t \cup \Gamma_2^t\right) = E\left(\Gamma_1^t\right) + E\left(\Gamma_1^t\right)$ is defined by [36]:

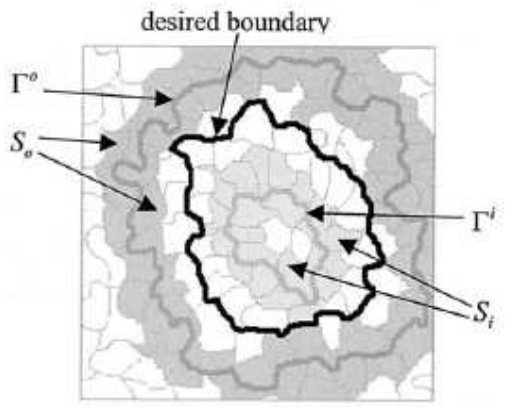


Figure 5.5: Region of interest (ROI) and initial snakes.

$$E\left(\Gamma_i^t\right) = \alpha_i E_{len}\left(\Gamma_i^t\right) + \beta_i E_\theta\left(\Gamma_i^t\right) + \gamma_i E_{ext}\left(\Gamma_i^t\right) + \delta_i E_{Area}\left(\Gamma_i^t\right), \quad i = 1, 2, \tag{5.58}$$

where:

$$E_{len}\left(\Gamma_i^t\right) = \sum_{j=0}^{N_i-1} \left|v_{i,j}^t - v_{i,j}^t\right|, \tag{5.59}$$

$$E_{\theta}\left(\Gamma_i^t\right) = \sum_{j=0}^{N_i-1} \cos^{-1}\left(\frac{u_{i,j}^t \cdot u_{i,j+3}^t}{\left|u_{i,j}^t\right|\left|u_{i,j+3}^t\right|}\right), \tag{5.60}$$

$$E_{ext}\left(\Gamma_i^t\right) = \sum_{j=0}^{N_i-1} \left|\nabla I\left(v_{i,j}^t\right)\right|, \tag{5.61}$$

$$E_{Area}\left(\Gamma_i^t\right) = Area\left(S_{io}^t\right), \tag{5.62}$$

where $v_{i,j}^t$ means the jth snaxel of the ith snake at time t, $u_{i,j}^t = v_{i,j}^t - v_{i,j-3}^t$, S_{io}^t denote the set of cells enclosed by the inner and outer snakes at the deformation step t, and $\alpha_i, \beta_i, \gamma_i, \delta_i, i = 1, 2$, are real parameters that controls the influence of each term in the expression (5.58).

The first energy term E_{len} is a slight modification of expression (3.3), in discrete form, which gives the perimeter of the contour Γ_j^t. If we take $\alpha_1 < 0$ and $\alpha_2 > 0$, then, minimizing this energy would force the inner snake into expanding outward and the outer snake into contracting. The term E_θ approximates the curvature along the snake and aims to control the smoothness of the contour. The Figure 3.1 helps to compare the geometry corresponding to E_θ with the one related to E_{int} in equation (5.25). From that figure and the definition of vector V_1 in Figure 3.1, a straightforward application of the Law of Cosines gives:

$$(2V_1)^2 = b^2 + c^2 + 2bc\cos\psi, \tag{5.63}$$

$$a^2 = b^2 + c^2 + 2bc\cos\omega. \tag{5.64}$$

Then, a simple algebra renders:

$$\cos\omega = \frac{a^2}{4bc}\left(1 - \frac{(2V_1)^2}{a^2}\right) = \frac{a^2}{4bc}\left(1 - E_{int}\left(v_{i-1}, v_i, v_{i+1}\right)\right), \tag{5.65}$$

where $E_{int}\left(v_{i-1}, v_i, v_{i+1}\right)$ is defined in equation (5.25). This expression shows the relationships between ω and $E_{int}\left(v_{i-1}, v_i, v_{i+1}\right)$. We observe that ω has a global

minimum when $V_1 = 0$ ($\omega(0) = 0$) and, if we keep a unchanged and scale b, c, we notice that $\omega \to \pi$ if $b, c \to +\infty$ (curvature infinite). The same happens for $E_{int}(v_{i-1}, v_i, v_{i+1})$. Therefore both E_{int} and ω incorporates a measure of local smoothness.

The third energy E_{ext} is the external energy defined from expression (3.9). The last energy, E_{area}, is the area covered by the cells in S_{io}^t. When minimizing energy (5.58) this term will provide the attraction force to pull the inner and the outer snakes to each other.

The energies (5.59)-(5.62) may have different ranges. Therefore, the target function to be minimized is the normalized energy variation $\Delta E(\Gamma_i^t)$ rather than the $E(\Gamma_i^t)$ itself, that is:

$$\Delta E\left(\Gamma_1^t, \Gamma_2^t; \Gamma_1^{t+1}, \Gamma_2^{t+1}\right) = \tag{5.66}$$

$$\alpha_i \Delta E_{len} + \beta_i \Delta E_\theta + \gamma_i \Delta E_{ext} + \delta_i \Delta E_{Area}, \tag{5.67}$$

where ΔE_{len}, ΔE_θ, ΔE_{ext}, ΔE_{Area} have the general form:

$$\Delta E_\rho\left(\Gamma_1^t, \Gamma_2^t; \Gamma_1^{t+1}, \Gamma_2^{t+1}\right) = \tag{5.68}$$

$$\frac{\left[E_\rho\left(\Gamma_1^{t+1}\right) + E_\rho\left(\Gamma_2^{t+1}\right)\right] - \left[E_\rho(\Gamma_1^t) + E_\rho(\Gamma_2^t)\right]}{\left[E_\rho(\Gamma_1^t) + E_\rho(\Gamma_2^t)\right]}, \tag{5.69}$$

with $\rho \in \{len, \theta, ext, Area\}$ and α_i, β_i, γ_i and δ_i are the same of expression (5.58).

Before to proceed, we need other definitions. Let S_1^t be the set of cells inside the ROI that intersect the curve Γ_1^t (similarly for S_2^t). Besides, it is defined in [36] the operators Φ and Ψ such that:

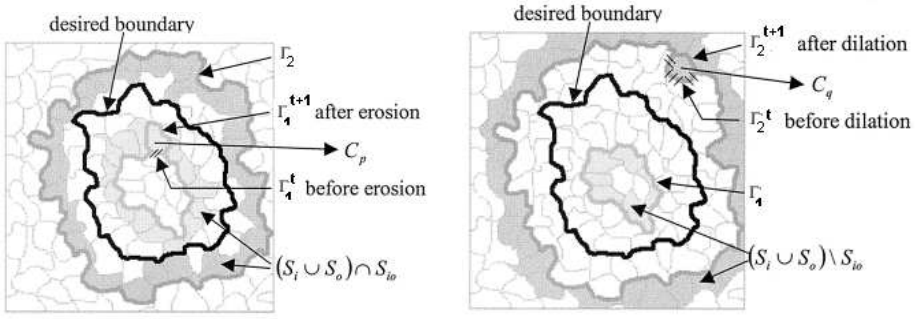
$$\Phi\left(S_{io}^t\right) = \Gamma_2^t, \quad and \quad \Psi\left(S_{io}^t\right) = \Gamma_1^t. \tag{5.70}$$

The deformation of the model is based on the two operators called **cell-erosion** and the **cell-dilation**. These operators are described on Figure 5.6. The former, denoted by CE, is defined as:

$$CE\left(S_{io}^t, C_p\right) = S_{io}^t - \{C_p\}, \quad C_p \in \left(S_1^t \cup S_2^t\right) \cap S_{io}^t. \tag{5.71}$$

The cell-dilation, denoted by CD, is defined by:

$$CD\left(S_{io}^{t}, C_p\right) = S_{io}^{t} \cup \{C_p\}, \quad C_p \in \left(S_1^t \cup S_2^t\right) - S_{io}^{t}. \tag{5.72}$$



(a) (b)

Figure 5.6: (a)Erosion operator. (b) Dilation operation.

The cell-based dual snakes evolution is the greedy procedure that aims to minimize the normalized energy variation defined in expression (5.67). It is based on the following algorithm to find the minimum of the energy variation given by expression (5.67).

1. Initialization: define S_{io}^{0}, $\Phi\left(S_{io}^{0}\right) = \Gamma_2^0$ and $\Psi\left(S_{io}^{0}\right) = \Gamma_1^0$ and set $t = 0$.
2. While $S_{io}^{t} \neq \emptyset$,

$$C_e = \underset{C_p \in \left(S_1^t \cup S_2^t\right) \cap S_{io}^t}{\arg\min} \left\{\Delta E_{erode} = \Delta E\left(\Gamma_1^t, \Gamma_2^t; \Phi\left(CE\left(S_{io}^{t}, C_p\right)\right), \Psi\left(CE\left(S_{io}^{t}, C_p\right)\right)\right)\right\}, \tag{5.73}$$

$$C_d = \underset{C_p \in \left(S_1^t \cup S_2^t\right) - S_{io}^t}{\arg\min} \left\{\Delta E_{dilate} = \Delta E\left(\Gamma_1^t, \Gamma_2^t; \Phi\left(CD\left(S_{io}^{t}, C_p\right)\right) \cup \Psi\left(CD\left(S_{io}^{t}, C_p\right)\right)\right)\right\}, \tag{5.74}$$

2.1) If $\Delta E_{erode} \leq \Delta E_{dilate}$, $S_{io}^{t+1} = CE\left(S_{io}^{t}, C_e\right)$. Else, $S_{io}^{t+1} = CD\left(S_{io}^{t}, C_d\right)$.

2.2) $\Gamma_2^{t+1} = \Phi\left(S_{io}^{t+1}\right)$ and $\Gamma_1^{t+1} = \Psi\left(S_{io}^{t+1}\right)$.

Once the boundary is extracted based on this algorithm, some kind of smoothing process may be applied in order to improve it. This stage can be performed

by an usual snake model, that is, the dual result is used to initialize a parametric (single) snake model. Other possibility, also discussed in [36], would be spline interpolation.

The Figure 5.7 pictures the stages of the method for breast ultrasound segmentation. Figure 5.7.a pictures the original image with a benign lesion. This input image is then filtered by a gaussian kernel and its low frequency version is presented on Figure 5.7.b. The gradient map obtained by using the Sobel edge detector is shown in Figure 5.7.c and the cells generated by the watershed algorithm are pictured on Figure 5.7.d. The initialization of the method and the obtained result (white contour) are shown on Figures 5.7.e and 5.7.f, respectively.

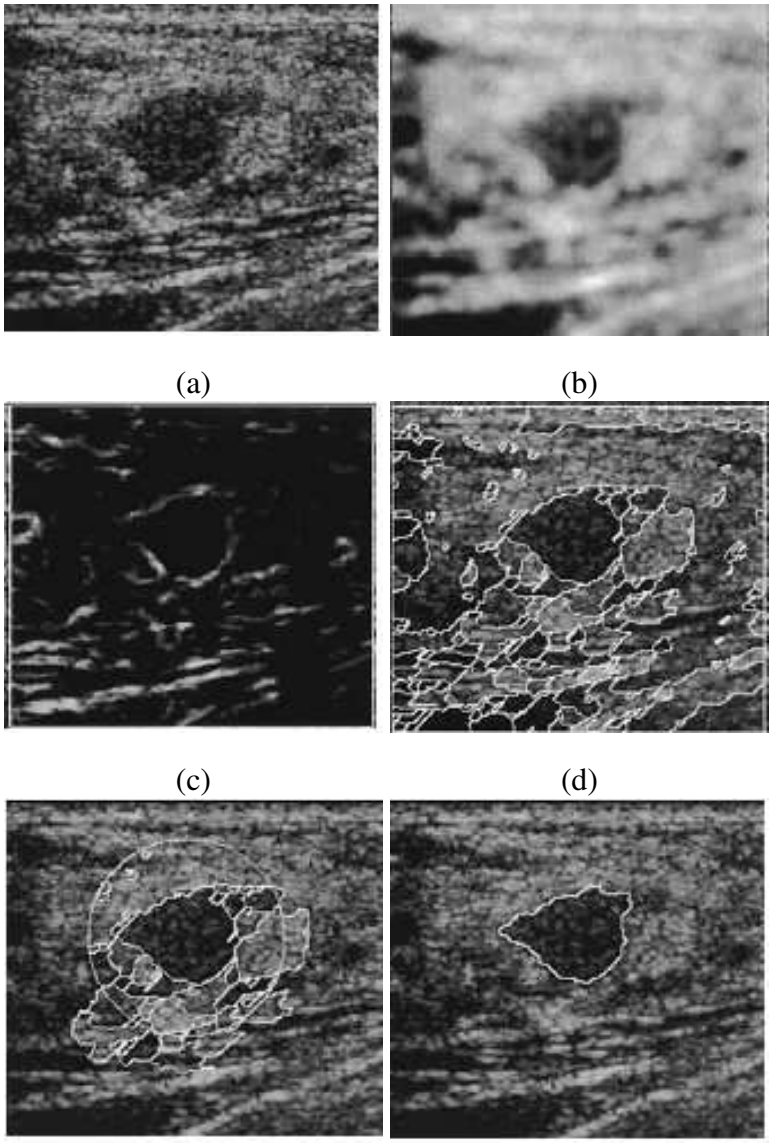


Figure 5.7: (a)Original breast ultrasound image. (b)Smoothed image. (c) Sobel edge detector result. (d) Cells generated by the watershed algorithm. (e)Initial snakes (white contours). (f) Final result.

Chapter 6

Snake Models Based on Local Deformations

6.1 Introduction

In this section we focus in discrete models that evolves through an evolution equation that is not explicitly derived from an energy functional. The evolution equation is based on local information about model geometry and the image field.

6.2 T-Snakes Model

It is important to observe that the space Ad in expression (3.2) 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 (7.15). Among the proposals for incorporating topological capabilities to parametric snake models [10, 11, 12, 13], the T-Snakes approach [47] has the advantage of been a general one, in the sense that the same formulation can be used for $2D$ and $3D$, for both splits and merges.

The T-Snakes approach is composed basically by four components [14]: (1) a simple CF-triangulation of the image domain; (2) projection of each snake over the grid; (3) a binary function called characteristic function χ, defined on the grid

nodes, which distinguishes the interior from the exterior of a snake; (4) a discrete snake model.

To clear the ideas, consider the characteristic functions (χ_1 and χ_2) relative to the two contours pictured in Figure 6.1. The vertices marked are those where $\max\{\chi_1, \chi_2\} = 1$. Observe that the merge of the curves belongs to the triangles in which the characteristic function changes value.

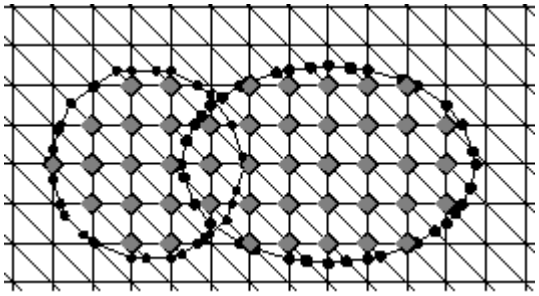


Figure 6.1: Two snakes colliding with the inside grid nodes and snake points (snaxels) marked.

Thus, from the points obtained in the step (2), we can choose a set of N points $\{v_i = (x_i, y_i), i = 0, ..., N-1\}$ to be connected to form a closed contour (T-Snake).

In [48] we evolve a T-Snake based on a tensile force (B_i), given by equation (3.26), an external (image) force (f_i), defined by expression (3.16), and a normal (balloon-like) force (F_i), defined as follows:

$$F_i = k\,(sign_i)\,\overrightarrow{n}_i, \tag{6.1}$$

where $\overrightarrow{n}_i$ is the normal at the snaxel v_i, k is a force scale factors, $sign_i = 1$ if $I(v_i) \geq T$ and $sign_i = 0$ otherwise (T is a threshold for the image I). The utility of the balloon-like force given by expression (6.1) is two-fold. Firstly, it avoids the unwanted contraction effect of the tensile force due to the inflation (see [44] for details). Secondly, this force pushes the model towards the object(s) of interest, characterized by the threshold T. The external force f_i^t given by expression (3.16) attracts the snake to object boundaries in the image. We update the T-Snake position according to the evolution equation defined by expression (3.25), which can be rewritten as:

$$v_i^{(t+\Delta t)} = v_i^t + \Delta t \left(B_i^t + F_i^t + f_i^t \right). \tag{6.2}$$

The T-Snakes model incorporates also an entropy condition: "once a node is burnt (passed over by the snake) it stays burnt " [14]. The termination condition defined by expression (3.27) is not efficient for the T-Snakes model due to its reparameterization process which is based on the projection of the snake [42, 46]. Therefore, a specific termination condition is defined based on the number of deformations steps (temperature) that a triangle was cut by a T-Snake. A T-Snake is considered to have reached its equilibrium state when the temperature of all the snaxels fall bellow a preset value (called "freezing point" in the T-Snakes literature [14, 47]).

6.3 Dual-T-Snakes Algorithm

The key idea behind this method is to explore the T-Snakes framework to propose a generalized dual active contour model: one T-Snake contracts and splits from outside the targets and another ones expand from inside the targets.

To make the outer snake to contract and the inner ones to expand we assign an inward normal force to the first and an outward normal force to the others according to expressions (6.1). Also, to turn the T-Snakes evolution interdependent we use the image energy and an affinity restriction.

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

$$E_{outer} = \sum_{i=0}^{N-1} \left(- \left\| \nabla I \left(v_i \right) \right\|^2 \right) / N, \tag{6.3}$$

$$E_{inner} = \frac{1}{m} \left(\sum_{k=0}^{m} \left(\sum_{i=0}^{N_k - 1} \left(- \left\| \nabla I \left(v_i \right) \right\|^2 \right) / N_k \right) \right), \tag{6.4}$$

where m is the number of inner curves, N, N_k are the number of snaxels of the outer snake and of the inner snake k, respectively. Expressions (6.3)-(6.4) can be obtained from equation (3.23) if we set $h_i = 1/N$ for the outer snake and proceed similarly for the inner ones. The normalization is necessary in order to

be compared. Otherwise, the snake energy would be a decreasing function of the number of snaxels and comparisons would not make sense.

Following the dual approach methodology [17], if $E_{inner} > E_{outer}$ an inner curve must be chosen. To accomplish this, we use an affinity operator which estimates the pixels of the image most likely to lie on the boundaries of the objects. Based on this operator, we can assign to a snaxel the likelihood that it is close to a boundary. That likelihood is thresholded to obtain an affinity function that assigns to the snaxel a 0-1 value: "0" for the snaxels most likely to lie away from the target boundaries and "1" otherwise.

Then, the inner curve with highest number of snaxels with affinity function value null is chosen. If $E_{outer} > E_{inner}$ the outer snake is evolved if the corresponding affinity function has null entries.

Also, the balance between the energy/affinity of the outer and inner snakes allows to avoid local minima. For instance, if a T-Snake has been frozen, we can increase the normal force at the snaxels where the affinity function is zero, that is, we add a driving force only to the snaxels most like to lie far from the boundary. The self-intersections that may happen during the evolution of a snake when increasing the normal force are naturally resolved by the T-Snakes model. This is way we can use that added normal force to play the role of the driving force used by Gunn and Nixon (avoiding the matching problem required in [17]).

To evaluate similarity between two contours, we use the difference between the characteristic function of the outer snake and the characteristic functions of the inner ones (Characteristic_Diff). For example, in the case of the CF triangulation of the Figure 6.1 we can stop the motion of all snaxels of an inner snake inside a triangle σ if any of its vertex $v \in \sigma$ has the two following properties: (a). All the six triangles adjacent to v have a vertex where Characteristic_Diff=0; (b). One of these triangles is crossed by the outer contour.

The freezing point (section 6.2) is used to indicate that a T-Snake has found an equilibrium position. In what follows, we call Dual Snake a list of T-Snakes where the first one is an outer contour and the other ones are inner contours. The algorithm can be summarized as follows:

The experience with this method shows that it is very useful to reduce the search space. So, we proposed in [22] a two stage segmentation approach: (1) the region of interest is reduced by the Dual-T-Snakes; (2) a global minimization

Algorithm 2 Dual-T-Snakes

Put all the dual snakes into a queue.

repeat

Pop out a dual snake from the queue;

Use the energies (equations (6.3) and (6.4)) and the affinity function to decide the snake to be processed;

if all snaxels of that snake are frozen **then**

repeat

increase the normal force at those with affinity zero

until the snake energy starts decreasing.

Remove that added normal force;

repeat

Evolve the snake

until the temperature of all snaxels falls bellow the freezing point;

Analyze the Characteristic_Diff of the current snake;

if the snake being processed is close to a snake of the other type (inner/outer) **then**

remove the dual snake from the queue.

else

mount the resulting dual snake(s) and go to the beginning.

until the queue is empty

technique, the Viterbi algorithm [33, 42], is used to find the object boundaries (see section 5.3). In fact, the Viterbi algorithm was also used in [38] and sometimes it is called non-evolutionary dual model, in the sense that it is not based on a curve evolution [61].

The following example shows the application of the segmentation framework that combines the Dual-T-Snakes and the Viterbi algorithm [22, 42] for a cell image. The Figure 6.2.a shows a blood cell obtained by an electronic microscope technique. When pass-band filter is applied, we get an edge map resembling a ribbon whose thickness depends on the kernel size of the used filter (Figure 6.2.b). That is an ideal situation for applying Dual-T-Snakes plus Viterbi because, firstly, the former extracts the ribbon (Figure 6.2.c). Then, the later is applied to the original image to give the final result (Figure 6.2.d).

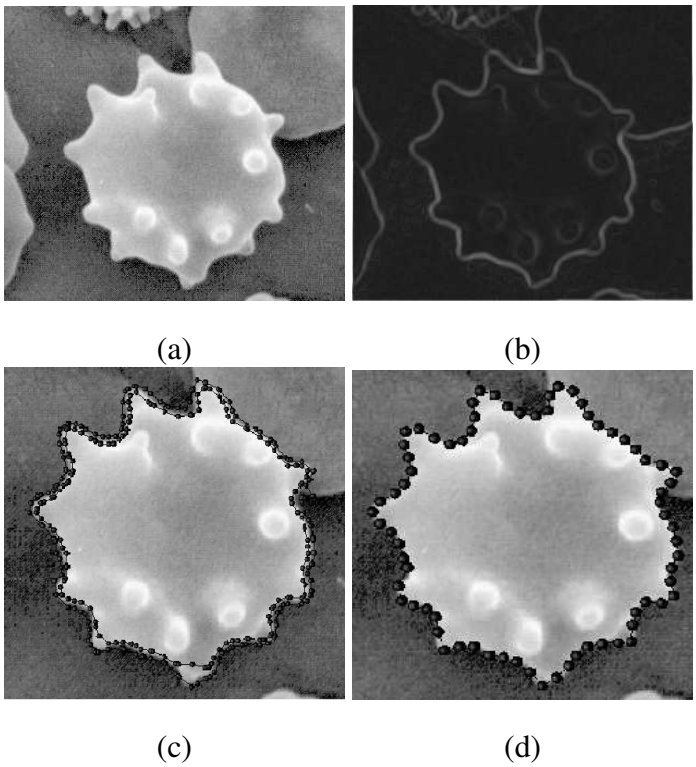


Figure 6.2: (a)Image to be processed. (b)Band-Pass filtered image. (c)Dual-T-Snakes solution. (d)Viterbi solution.

6.4 Neural Nets for Initialization

Neural Networks have been used for instantiating deformable models for face detection [105] and handwritten digit recognition tasks [81] (see also [97] and references therein). To the best of our knowledge, there is no references using neural nets to initialize deformable models for medical images. However, the network system proposed in [106], which segments MR images of the thorax, may be closer to this proposal.

In this method each slice is a grey level image composed of (256×256) pixels values and is accompanied by a corresponding (target) image containing just the outline of the region. Target images were obtained using a semi-automatic technique based on a region growing algorithm. The general idea is using a multilayer perceptron (MLP), where each pixel of each slice is classified into a contour-boundary and non-contour-boundary one.

The inputs to the MLP are intensity values of pixels from a (7×7) window centered on the pixel to be classified. This window size was found to be the smallest that enabled the contour-boundary to be distinguished from the others image's artifacts. The output is a single node trained to have an activation of 1.0 for an input window centered in the pixel of a contour-boundary, and 0.0 otherwise. The network has a single hidden layer of 30 nodes.

The network was trained using error back-propagation [107, 108] with weight elimination [109] to improve the network's generalization ability. The training data should be constructed interactively: a proportion of misclassified examples should be added to the training set and used for retraining. The process is initiated from small random selection of contour-boundary and non-contour-boundary examples, and should be terminated when a reasonable classification (on a given slice) is achieved.

The MLP classified each pixel independently of the others, and therefore has no notion of a closed contour. Consequently, the contour-boundaries it produces are often fragmented and noisy (false negatives and false positives respectively). Then, with this initial set of points classified as contour-boundaries, a deformable model is used to link the boundary segments together, while attempting to ignore noise.

In [106] it is used the *Elastic Net* algorithm. This technique is based on the following equations:

$$\Delta u_{j,l}^{t+1} = \alpha \sum_{i=1}^{N} G_{ij} \left(p_{i,l} - u_{j,l}^{t} \right) + K\beta \left(u_{j+1,l}^{t} - 2u_{j,l}^{t} + u_{j-1,l}^{t} \right), \tag{6.5}$$

$$\Delta u_{j,l}^{t+1} = K\gamma \left(u_{j,l+1}^{t+1} - 2u_{j,l}^{t+1} + u_{j,l-1}^{t+1} \right), \tag{6.6}$$

where $\Delta u_{j,l}^{t+1}$ is a inter-slice smoothing force, K is a simulated annealing term, α, β, γ are pre-defined parameters and G_{ij} is a normalized gaussian that weights the action of the force that acts over the net point $u_{j,l}$ due to edge point $p_{i,l}$ (l is the slice index).

The deformable model initialization is performed by using a large circle encompassing the lung boundary in each slice. This process can be improved by using the training set.

As an example, let us consider the work [81] in handwritten digit recognition. In this reference, each digit is modelled by a cubic B-spline whose shape is determined by the positions of the control points in the object-based frame. The models have eight control points, except for the one model which has three, and the seven model which has five. A model is transformed from the object-based frame to the image-based frame by an affine transformation which allows translation, rotation, dilation, elongation, and shearing. The model initialization is done by determining the corresponding parameters. Next, model deformations will be produced by perturbing the control points away from their initial locations.

There are 10 classes of handwritten digits. A feedforward neural network is trained to predict the position of the control points in a normalized 16×16 grey level image. The network uses a standard three layer architecture. The outputs are the location of the control points in the normalized image. By inverting the normalization process, the positions of the control points in the unnormalized image are determined. The affine transformation corresponding to these image can then be determined by running a special search procedure.

Chapter 7

Continuous and Parametric Snake Models

7.1 Introduction

In this Chapter, we describe some continuous snake models in which the deformable curve is represented in a parametric form. Following the taxonomy of Figure 3.2, we must consider models that are based on Euler-Lagrange equations and front propagation techniques. The original snake model, presented on section 3.2 is a technique in this class.

7.2 Scale-Space Methods and Snakes

The external potential in expression (3.8) depends on the derivatives of the image field. However, the images are, in general, irregular and noisy fields, as we can see on Figure 7.1 for the $1D$ case. Therefore, from the numerical viewpoint, some regularization process is necessary. The linear diffusion (Gaussian blurring), discussed on section 4.2, is the straightforward method to perform this task.

The application of the Gaussian Kernel was the starting point for the parametric multiscale representation of images [110]. The central idea of such methods is to consider the original image as a member of an image family $I(x, y, t)$ generated

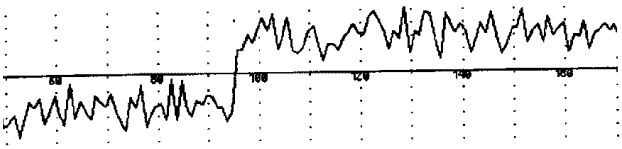


Figure 7.1: Irregular and noisy signal f.

by the convolution of the original image $I_0(x, y) = I(x, y, 0)$ with the Gaussian kernel [111]:

$$I\left(x, y, t\right) = G_t * I_0, \tag{7.1}$$

where:

$$G_{\sigma(t)}\left(x, y\right) = \frac{1}{2\pi\sigma\left(t\right)^2} \exp\left(-\frac{1}{2\sigma\left(t\right)^2}\left(x^2 + y^2\right)\right) \tag{7.2}$$

is the $2D$ Gaussian kernel. The family $I\left(x, y, t\right)$ in expression (7.1) is the fundamental solution of the equation (4.1) if $c\left(x, y, t\right)$ is a constant.

Such formulation has the following properties: (a) Linear; (b) Obeys a causality principle (no artifacts are generated along the time t in the above scheme); (c) It recovers the original signal if $\sigma \to 0$ because

$$\lim_{\sigma \to 0} G_\sigma\left(x, y\right) = \delta\left(x, y\right), \tag{7.3}$$

where $\delta\left(x, y\right)$ is the Dirac delta.

Therefore, we can show that the gradient operator applied over the smoothed field is well defined. In fact, let us consider a (generic) differentiable filter s_σ and a scalar field Φ, where σ is a real parameter that controls the filter support. The smoothed version $\widetilde{\Phi}$ of Φ, given by the convolution:

$$\widetilde{\Phi} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \Phi\left(v, w\right) s_\sigma\left(u_1 - v, u_2 - w\right) dvdw, \tag{7.4}$$

is a differentiable function respect to x and y because, given a differential operator:

$$D = \sum_{i,j} b_{ij} \frac{\partial^n}{\partial u_1^i \partial u_2^j}, \quad n = 1, 2, ... \tag{7.5}$$

we can apply it to the field $\widetilde{\Phi}$ by:

$$D\left(\widetilde{\Phi}\right) = \int_{-\infty}^{+\infty} \int_{-\infty \Phi}^{+\infty} \Phi(v, w) \left[\sum_{(i,j) \in N_n^2} b_{ij} \frac{\partial^n}{\partial u_1^i \partial u_2^j} s_\sigma (u_1 - v, u_2 - w) \right] dv dw. \tag{7.6}$$

Therefore, we can compute $D\left(\widetilde{\Phi}\right)$ in only one step by just taking the convolution of the original field Φ with the derivative of the kernel s. Likewise the Gaussian kernel, this kernel must obey the properties (a)-(c) listed above.

Expression (7.6) is the starting point to generalize the idea of (linear) multiscale representation of images. The idea is represented on Figure 7.2.

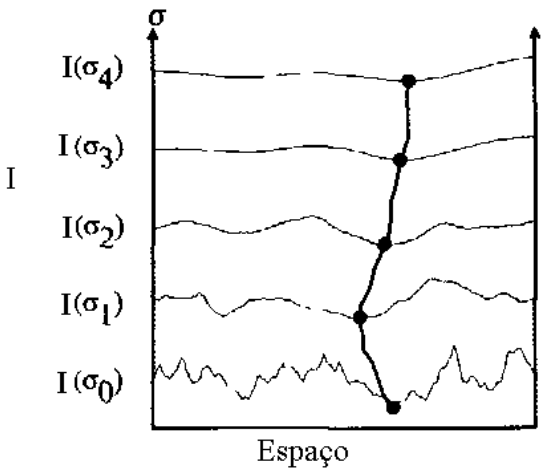


Figure 7.2: Tracking in the scale space.

In this figure, we picture a signal I and a sequence of low pass versions, obtained by convolving the original signal with the kernel s_σ, through expression (7.4), at scales $\sigma_4 > \sigma_3 > ... > \sigma_0$. The key idea is, to identify the structure of interest in the coarser scale σ_4, and then to track it over the finer scales until the finest one σ_0.

7.3 Euler-Lagrange Snakes

The process of minimizing the functional given in (3.2) can be viewed from a dynamic point of view by using the Lagrangian mechanics [15, 118]. 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:

$$c(s,t) = (x(s,t), y(s,t)), \tag{7.7}$$

with a mass density μ and a damping density λ. Therefore, the curve has a potential energy E_p, defined by the expression (3.2), a kinetic energy T which depends on the velocity [6, 112]:

$$E_p(c) = \int_c \left(\omega_1 \|c'(s)\|^2 + \omega_2 \|c''(s)\|^2\right) ds + \int_c P(c(s))\, ds, \tag{7.8}$$

$$T = \mu \int_c \left\| \frac{\partial c}{\partial t} \right\|^2 ds. \tag{7.9}$$

The damping forces are derived from a generalized potential U_{at} which is a function of the curve velocity [112]:

$$\frac{\partial U_{at}}{\partial t} = -\frac{1}{2}\lambda \left\| \frac{\partial c}{\partial t} \right\|^2, \tag{7.10}$$

Therefore, the Lagrangian of the problem can be written as [112]:

$$L\left(c, \dot{c}, c'.c''\right) = T - E_p - U_{at}, \tag{7.11}$$

Once we have two independent parameters s and t, the functional J to be optimized has the form [112]:

$$J = \int_{t_1}^{t_2} \int_0^1 L ds dt \tag{7.12}$$

which generates the following Euler-Lagrange equations [112, 92]:

$$\frac{\partial L}{\partial x} - \frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial}{\partial s}\left(\frac{\partial L}{\partial x'}\right) + \frac{\partial}{\partial s^2}\left(\frac{\partial L}{\partial x''}\right) = 0, \tag{7.13}$$

$$\frac{\partial L}{\partial y} - \frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{y}}\right) - \frac{\partial}{\partial s}\left(\frac{\partial L}{\partial y'}\right) + \frac{\partial}{\partial s^2}\left(\frac{\partial L}{\partial y''}\right) = 0. \tag{7.14}$$

By substituting expressions (7.8)-(7.10), we obtain the equations of motion for the snake [15, 46]:

$$\mu\frac{\partial^2 c}{\partial t^2} + \lambda\frac{\partial c}{\partial t} - w_1\frac{\partial^2 c}{\partial s^2} + w_2\frac{\partial^4 c}{\partial s^4} - \gamma\nabla P = 0, \tag{7.15}$$

where the first two terms (from left to right) represent the inertial and damping forces while the third and fourth terms give the forces related to the internal energies (we are supposing constant parameters). Now, the non-invariance of the internal energy under affine transformations can be seen from a dynamical point of view: the internal forces in equation (7.15) have an inward normal component, observed in expression (3.7), which tends to contract the snake in order to push the model towards the global minimum of $w_1 E_{ten} + w_2 E_{rig}$ [19, 20, 45]. The last term in equation (7.15) is the external force due to the external potential P defined by expressions (3.8)-(3.9).

Given an initial curve ($c\,(s, 0)$) and its velocity, the equation (7.15) can be solved to find the minimum of the potential energy defined by expression (7.8). Such equilibrium position is achieved when the internal and external forces balance and the contour comes to rest, which implies that:

$$\partial c/\partial t = \partial^2 c/\partial t^2 = 0. \tag{7.16}$$

In general, snake models do not use the inertia term, that means, $\mu = 0$, in order to avoid oscillations nearby the target (see [118] for details). Besides, the rigidity in expression (7.15) makes the snake too rigid and may be not efficient to get finer details. Therefore, we set $\mu = w_2 = 0$ in expression (7.15) and then divide the resulting equation by the damping density λ in order to discard one more parameter. The obtained model has the evolution equation given by:

$$\frac{\partial c}{\partial t} - w_1\frac{\partial^2 c}{\partial s^2} - \gamma\nabla P = 0. \tag{7.17}$$

which is equal to expression (3.19), of section 3.2.

7.4 Snakes in Finite Dimensional Spaces

In this discussion we are supposing deformable curves that can be represented in the form:

$$c(s) = \sum_{i=0}^{N} q_i f_i(s) = B^T Q, \tag{7.18}$$

where $B = (f_0, f_1, ..., f_N)^T$ is an array of linearly independent functions and Q is a column vector of points of the plane which are called the control points of the curve. The most common examples of these deformable models are the D-NURBS [119] and Active Splines [6] which are based on piecewise polynomial functions belonging to a finite dimensional space of square integrable functions.

The instantaneous configuration of the system is given by a point in a configuration space of dimension $(2N + 2)$, whose (*generalized*) coordinates are given by the coordinates of the $N + 1$ points q_i in the Q vector.

As the curve is deformed, the control points get move, so we have a velocity vector $\dot{Q}$ associate with Q. The complete state of the system (snake) is defined by a pair $\left(Q, \dot{Q}\right)$.

An advantage of using curves of the form (7.18) is that the motion equations in the Q space (Euler-Lagrange equations) are ordinary differential equations. To see this, let's compute the energies given by expressions (7.8)-(7.10), but now considering that the curve $c(s,t)$ has the form of expression (7.18):

Kinetic Energy

$$T\left(\dot{c}\right) = \frac{1}{2}\int \mu \left\| \frac{\partial c}{\partial t} \right\| ds = \frac{\mu}{2}\dot{Q}^T M_0 \dot{Q}. \tag{7.19}$$

where $M_0 = \int BB^T ds$ and μ is de linear mass.

Dissipation

$$\frac{\partial U_{at}}{\partial t}\left(\dot{c}\right) = -\frac{\gamma}{2}\dot{Q}^T M_0 \dot{Q}. \tag{7.20}$$

where γ is the constant damping density.

Elastic Energy

$$E_{int} = \omega_1 Q^T M_1 Q + \omega_2 Q^T M_2 Q. \tag{7.21}$$

where ω_1 and ω_2 are the tension and rigidity parameters, already defined on section 3.2 and:

$$M_1 = \int \frac{dB}{ds} \cdot \frac{dB^T}{ds} ds, \qquad M_2 = \int \frac{d^2B}{ds^2} \cdot \frac{d^2B^T}{ds^2} ds. \tag{7.22}$$

External Energy

$$E_{ext} = \int P\left(c\left(s,t\right)\right) ds. \tag{7.23}$$

Potential Energy

$$E_p = E_{int} + E_{ext}. \tag{7.24}$$

The corresponding Lagrangian has the form [112]:

$$L\left(c, \dot{c}, c'.c''\right) = T - E_p - U_{at}. \tag{7.25}$$

The Euler-Lagrange Equations derived from the Hamilton's Principle [112] for the system described by (7.19)-(7.24) compose the snake model that we will study in this section. The Euler-Lagrange Equations for the snake can be summarized as [112, 6]:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{Q}}\right) - \frac{\partial T}{\partial Q} - \frac{\partial}{\partial \dot{Q}}\left(\frac{\partial U_{at}}{\partial t}\right) + \frac{\partial E_p}{\partial Q} = 0 \tag{7.26}$$

By substituting (7.19)-(7.24) in this expression we obtain:

$$\mu\ddot{Q} + \gamma\dot{Q} + M_0^{-1}\left(\omega_1 M_1 + \omega_2 M_2\right) Q = M_0^{-1} F, \tag{7.27}$$

where:

$$F = -\frac{1}{2}\int \frac{\partial P\left(c\left(s,t\right)\right)}{\partial Q} ds. \tag{7.28}$$

The parameters $(\mu, \gamma, \omega_1, \omega_2)$ must be chosen in advance and are essential for the behavior of the model. We have also to constrain this system to initial conditions to have a unique solution.

7.4.1 Numerical Analysis

The equation (7.27) do not have in general analytical solution so we have to use a numerical approach to solve it with the desired precision. The finite differences scheme we use is similar to the one applied on section 3.2.1. It is given by the following approximations of the first and second time derivatives:

$$\dot{Q}\left(t-\tau\right) \approx \frac{1}{2\tau}\left[Q\left(t\right) - Q\left(t-2\tau\right)\right], \tag{7.29}$$

$$\ddot{Q}\left(t-\tau\right) \approx \frac{1}{\tau^2}\left[Q\left(t\right) - 2Q\left(t-\tau\right) + Q\left(t-2\tau\right)\right]. \tag{7.30}$$

with analogous expressions for space derivatives.

Through these expressions we can discretise the equation (7.27) in the form:

$$AQ^{t+\tau} = B^{t,t-\tau}. \tag{7.31}$$

where:

$$A = \left[\left(\frac{\mu}{\tau^2} + \frac{\gamma}{2\tau}\right) M_0 + K\right], \tag{7.32}$$

$$B^{t,t-\tau} = 2F\left(\bar{t}-\tau\right) + \left(\frac{2\mu}{\tau^2}\right) M_0 Q^t - \left(\frac{\mu}{\tau^2} - \frac{1}{2\tau}\right) M_0 Q^{t-\tau}, \tag{7.33}$$

Let's now study the properties of this scheme. For a numerical scheme gives coherent results it has to have the properties of consistence, stability and convergence [120]. The consistence is straightforward verified due the expressions (7.29)-(7.30).

For the stability condition of the scheme let's first suppose the Hessian of F is limited. So, the analysis of the stability can be carried out by the conditioning of matrix A. If A is well-conditioned then we have a guarantee of stability for the scheme.

Following the methodology found in [15], let's consider the condition number of the matrix A in 7.32. Supposing A is non singular, this number is given in 2-norm by [118]:

$$\kappa_2\left(A\right) = 1 + \frac{\lambda_{\max}\left(M_0\right) - \lambda_{\min}\left(M_0\right)}{\lambda_{\min}\left(M_0\right)} + \frac{\lambda_{\max}\left(M_0\right)}{\beta}, \tag{7.34}$$

where:

$$\beta = \left(\frac{\mu}{\tau^2} + \frac{\gamma}{2\tau}\right). \tag{7.35}$$

From (7.34)-(7.35) we see that the higher β and the lower is the difference $\lambda_{\max}(M_0) - \lambda_{\min}(M_0)$ the better is the conditioning of matrix A ($\kappa_2(A) \rightarrow 1$) and consequently the stability of the linear system. From expression (7.35) it is clear that we can improve the stability by increasing μ and γ. So, by the Lax Equivalence Theorem [120], we can guarantee that the numerical scheme is convergent.

Despite of the gain of stability, we have to be careful with large values of μ and γ because they can slowing down the convergence rate. In [118] we address this drawback in the context of dynamical system. Specifically, we show that the parameters of the model have to be chosen in a way that the border we seek is an attractor or stable focus in the topology of the phase space defined by $\left(Q, \dot{Q}\right)$. A necessary condition to achieve this is that the energy functional is strictly convex in a neighborhood of the border, that means, if E_p has a local minimum in the border. As a consequence of this analysis, a local expression relating the dynamic parameters and the rate of convergence arises. Such results link the convexity analysis of the potential energy and the dynamic snake model, generalizing the results presented in [121].

7.5 Dual-Front Approach

In [24], a fast and flexible dual-front implementation of active contours is propose by iteratively dilating an initial curve to form a narrow region and then finding the new closest potential weighted minimal partition curve inside. The method is motivated by minimal path technique [26, 25]. In this method, given a potential $P > 0$, and a point p in the domain Ω, the minimal action map $U_0(p)$ is defined as:

$$U_0(p) = \min_{A_{p_0,p}} \int_{\Omega} \widetilde{P}(c(s))\, ds \tag{7.36}$$

where $\widetilde{P} = P + w$, with w been a constant, and $A_{p_0,p}$ is the set of paths connecting p_0 and p. Expression (7.36) gives the minimal energy integrated along the paths

between the starting point p_0 and any point p inside the domain Ω. Because the action map U_0 has only one minimum value at the starting point p_0 and is a convex function in Ω, it can be easily determined by solving the equation [26]:

$$|\nabla U_0| = \widetilde{P}, \quad and \quad U_0(p) = 0, \tag{7.37}$$

which becomes the Eikonal equation (expression (8.9)) if we set $\widetilde{P} = 1/V$. This equation can be efficiently solve by using the fast marching algorithm [27] revised on section 8.2. Equations (7.36)-(7.37) are the starting point for the dual-front technique [24]. So, given two points p_0, $p_1 \in \Omega$ the method computes the action maps $U_0(p)$ and $U_1(p)$, respectively, through the solution of expression (7.37), seeking for the points $p \in \Omega$ such that:

$$U_0(p) = U_1(p). \tag{7.38}$$

At these points, the level sets of the minimal action map U_0 meets the level sets of the minimal action map U_1 generating the Voronoi diagram of the image that decompose the whole image into two regions containing the points p_0 and p_1. We can generalize definition (7.36) for a set $X \subset \Omega$ through the expression:

$$U_X(q) = \min_{p \in X} U_p(q), \tag{7.39}$$

that mens, $U_X(q)$ is the minimal energy along the paths of the set $A_{p,q}$, where $p \in X$. Therefore, given two curves c_{in} and c_{out} bounding the search space called R_n in the Figure 7.3, and two potentials $\widetilde{P}_{in}$ and $\widetilde{P}_{out}$ that takes lower values near desired boundaries, the dual-front algorithm firstly computes the minimal action maps U_{in} and U_{out} until these two action maps meet each other. Then, the evolutions of the level sets of both the action maps stops and a minimal partition boundary is formed in the region R_n of the Figure 7.3. Mathematically, this boundary is the solution of the following equations:

$$|\nabla U_{in}| = \widetilde{P}_{in}, \quad with \quad U_{in}(c_{in}) = 0, \tag{7.40}$$

$$|\nabla U_{out}| = \widetilde{P}_{out}, \quad with \quad U_{out}(c_{out}) = 0, \tag{7.41}$$

$$U_{in}(p) = U_{out}(p). \tag{7.42}$$

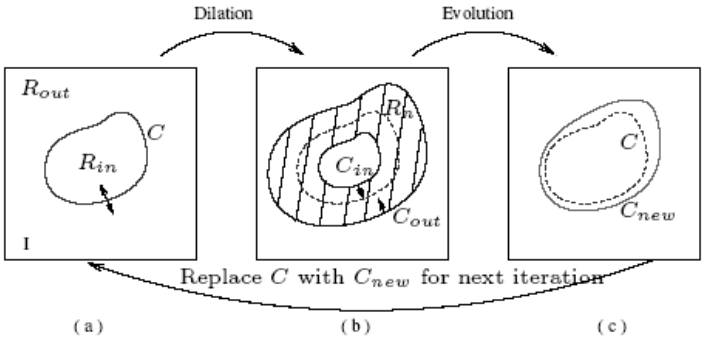


Figure 7.3: One interaction of the Dual-Front algorithm. (a)Initial curve c_n of the iteration n. (b)Search space defined through dilation of the initial curve c_n with bounds c_{in} and c_{out}. (c)Obtained solution C_{new}, the minimal partition curve. The curve c is replaced by the curve C_{new} to initialize the next iteration.

The dual-front approach is an iterative method which is picture on Figure 7.3. Firstly, the curves c_{in} and c_{out} are placed by user interaction or obtained by dilation of an initial curve, named by c_n in Figure 7.3. Then, the minimal partition boundary is computed by solving equations (7.40)-(7.42). To perform this task, the actions $U_{in}$and U_{out} are computed inside the region R_n, through expressions (7.40) and (7.41), respectively, until condition (7.42) is achieved. The obtained result, named by c_{n+1} will replace c_n for processing the next iteration. The method proceed until the distance between consecutive minimal partition curves is less than a pre-defined threshold δ, that means $d(c_n, c_{n+1}) < \delta$ (like in expressions (5.37)). The potentials $\widetilde{P}_{in}$ and $\widetilde{P}_{out}$ are defined in [24] using the following general expressions which integrates region based and the edge-based information.

$$\widetilde{P}_{in}(x,y) = w_{in}^{r} \cdot f\left(\left|I(x,y) - \mu_{in}\right|, \sigma_{in}^{2}\right) + w_{in}^{b} \cdot g(\nabla I) + w_{in}, \tag{7.43}$$

$$\widetilde{P}_{out}(x,y) = w_{out}^{r} \cdot f\left(\left|I(x,y) - \mu_{out}\right|, \sigma_{out}^{2}\right) + w_{out}^{b} \cdot g(\nabla I) + w_{out}, \tag{7.44}$$

where μ_{in}, σ_{in}^2 are the mean and variance of the image intensity inside region $(R_{in} - R_{in} \cap R_n)$, μ_{out}, σ_{out}^2 are the mean and variance of the image intensity

inside region $(R_{out} - R_{out} \cap R_{in})$, and $w_{in}^r, w_{in}^b, w_{in}$ are parameters to be set in advance (the same for $w_{out}^r, w_{out}^b, w_{out}$).

The Figure 7.4 shows an example of the application of the dual-front method for $2D$ human brain MRI image where the segmentation objective is to find the interface between the gray matter and the white matter. This example is interesting to observe the sensitivity of the method against the width of the search space, called active region in [24]. We observe that the obtained result was much better for the narrower search space than for the other ones. In this test, the potentials in expressions (7.43)-(7.44) are defined by setting $w_{in}^r = w_{out}^r = 1$, $w_{in}^b = w_{out}^b = 0.1$, $w_{in} = w_{out} = 0.1$, and f, g given by:

$$f(x,y) = |I(x,y) - \mu_{in}|, \tag{7.45}$$

$$g(x,y) = (1 + |\nabla I|)^2. \tag{7.46}$$

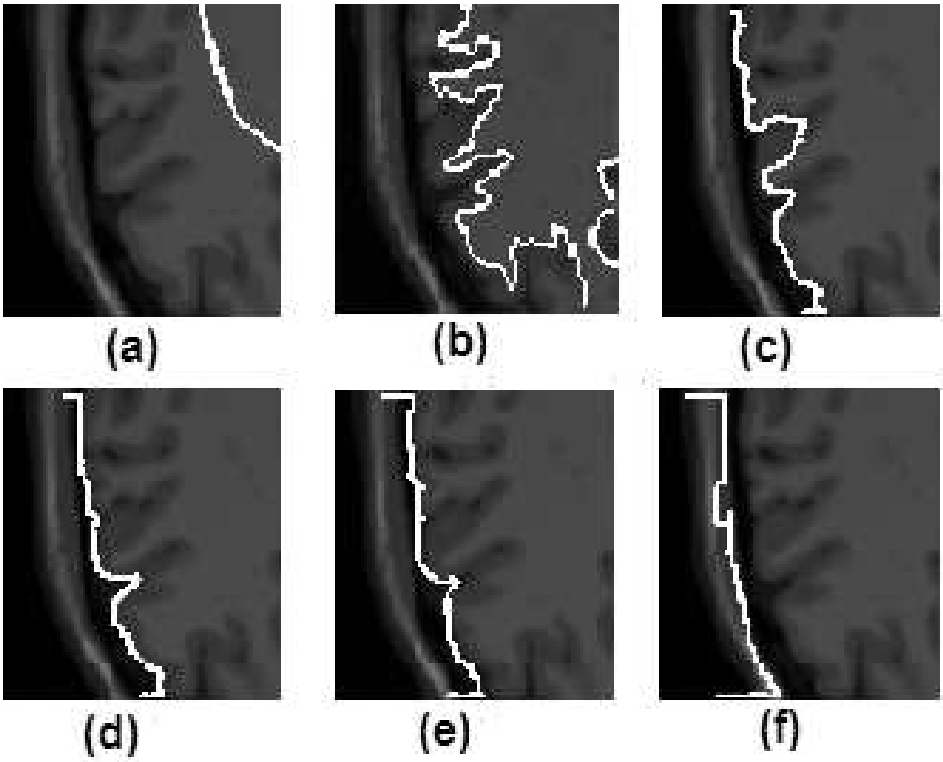


Figure 7.4: Sensitivity of the of the Dual-Front against different sizes of the active region (search space): (a) The original $2D$ human brain MRI image and the initial curve; (b) The corresponding edge map obtained through the gradient information; (c)-(g) Segmentation results obtained for a search space defined through morphological dilation of the initial curve with 5×5, 7×7, 11×11, 15×15, 23×23 pixels circle structuring elements, after 15 iterations.

Chapter 8

Implicit Models

8.1 Introduction

Implicit snake models, such as the formulation of *level set* proposed in [8], consist of embedding the snake as the zero level set of a higher dimensional function and to solve the corresponding equation of motion (see [9] for a review). Such methodologies are best suited for the recovery of objects with complex shapes and unknown topologies.

In section 8.2 we review the basic formulation of level set. In sections 8.4 and 8.5 we present applications of level set for dual and twin snakes, respectively.

8.2 Level Set

In this section we review some details of the level set formulation [8]. The main idea of this method is to represent the deformable surface (or curve) as a level set $\{x \in \Re^3 | G(x) = 0\}$ of an embedding function:

$$G : \Re^3 \times \Re^+ \rightarrow \Re, \tag{8.1}$$

such that the deformable surface (also called front in this formulation), at $t = 0$, is given by a surface S:

$$S(t=0) = \left\{ x \in \Re^3 | G(x, t=0) = 0 \right\}. \tag{8.2}$$

The next step is to find an Eulerian formulation for the front evolution. Following Sethian [8], let us suppose that the front evolves in the normal direction with velocity $\overrightarrow{F}$ that may be a function of the curvature, normal direction, etc.

We need an equation for the evolution of $G(x,t)$, considering that the surface S is the level set given by:

$$S(t) = \left\{ x \in \Re^3 | G(x, t) = 0 \right\}. \tag{8.3}$$

Let us take a point $x(t)$, $t \in \Re^+$ of the propagating front S. From its implicit definition given above we have:

$$G(x(t), t) = 0. \tag{8.4}$$

Now, we can use the Chain Rule to compute the time derivative of this expression:

$$G_t + F|\nabla G| = 0, \tag{8.5}$$

where $F = \|dx/dt\|$ is called the speed function and ∇ is the gradient operator, with respect to x. An initial condition $G(x, t=0)$ is required. A straightforward (and expensive) technique to define this function is to compute a signed-distance function as follows:

$$G(x, t=0) = \pm d, \tag{8.6}$$

where d is the distance from x to the surface $S(x, t=0)$ and the sign indicates if the point is interior (-) or exterior (+) to the initial front.

Finite difference schemes, based on an uniform grid, can be used to solve equation (8.5). The same entropy condition of T-Surfaces (once a grid node is burnt it stays burnt) is incorporated in order to drive the model to the desired solution (in fact, T-Surfaces was inspired on the level set model [46]).

In this higher dimensional formulation, topological changes can be efficiently implemented. Numerical schemes are stable, and the model is general in the sense that the same formulation holds for $2D$ and $3D$, as well as for merge and splits.

Besides, the surface geometry is easily computed. For example, the front normal ($\overrightarrow{n}$) and curvature (K) are given, respectively, by:

$$\overrightarrow{n} = \nabla G\left(x,t\right), \quad K = \nabla \cdot \left(\frac{\nabla G\left(x,t\right)}{\|\nabla G\left(x,t\right)\|} \right), \tag{8.7}$$

where the gradient (∇) and the divergent ($\nabla \cdot$) are computed with respect to x.

The update of the embedding function through expression (8.5) can be made cheaper if the narrow-band technique is applied. The key idea of this method comes from the observation that the front can be moved by updating the level set function at a small set of points in the neighborhood of the zero set instead of updating it at all the points on the domain (see [8, 49] for details).

To implement this scheme we need to pre-set a distance Δd to define the narrow band. The front can move inside the narrow-band until it collides with the narrow-band frontiers. Then, the function G should be reinitialized by treating the current zero set configuration as the initial one. The mathematical background behind this process is explained in section 8.2.

Also, this method can be made cheaper by observing that the grid points that do not belong to the narrow band can be treated as sign holders [8], following the same idea of the characteristic function used in section 6.2. This observation will be used in section 8.4.

Eikonal Equation and its Mathematical Solution

In this sub-section, we present the mathematical solution for solving the level set function with unity speed. Such a method is needed to compute the signed distance transform when the front collides with the narrow-band frontier. The key idea is to rewrite expression (8.5) in the form:

$$G_t = V(x,y)\left|\nabla G\right|, \tag{8.8}$$

where $V(x,y) = -F$ is the opposite of the speed function at a point $(x,y) \in \Re^2$. Let $T(x,y)$ be the time at which the front crosses the grid point (x,y). In this time, the surface $T(x,y)$ satisfies the Eikonal equation:

$$||\nabla T|| \; . \; V = 1. \tag{8.9}$$

By approximation, the solution to this equation would be given as:

$$max\ [\ max(D^{-x}\,T, 0), -min(D^{+x}\,T, 0)\ ]^2 +$$

$$max\ [\ max(D^{-y}\,T, 0), -min(D^{+y}\,T, 0)\ ]^2 = \frac{1}{{V^2}_{xy}}\,, \tag{8.10}$$

where ${V^2}_{xy}$ was the square of the speed at location (x, y), and $D^{-x}\,T$, $D^{+x}\,T$, $D^{-y}\,T$, $D^{+y}\,T$ are the backward and forward differences in space, given as:

$$\begin{cases} D^{+x}\,T = \frac{T(x+1,y)-T(x,y)}{2}, \\ D^{-x}\,T = \frac{T(x,y)-T(x-1,y)}{2}, \\ D^{+y}\,T = \frac{T(x,y+1)-T(x,y)}{2}, \\ D^{-y}\,T = \frac{T(x,y)-T(x,y-1)}{2}. \end{cases}$$

An analogous scheme can be obtained for $3D$. There are efficient schemes for solving Eikonal equation (8.9). For details, see [50, 51, 52]. We implemented equation (8.10) using Sethian's "fast marching" algorithm referred to in patent [53], discussed in the next sub-section.

Fast Marching Method for Solving the Eikonal Equation

The "fast marching method" (FMM) was used to solve the Eikonal equation, or a level set evolution with speed (F) whose sign does not change. Its main usage was to compute the signed distance transform from a given curve (say, one with speed = 1). This signed distance function was the level set function that was used in the "narrow band" algorithm. FMM can also be used for a simple active contour model if the contour only moves either inward (pressure force in terms of parametric snakes) or outward (balloon force in terms of parametric snakes). The FMM algorithm consisted of three major steps: (1) initialization stage, (2) tagging stage, and (3) marching stage. We will briefly discuss these three stages.

1. Initialization Stage:
   Let us assume that the curve cuts the grid points exactly, which means that it passed through the intersection of the horizontal and vertical grid lines. If the curve did not pass through the grid points, then we found where the curve intersected the grid lines using the simple method recently developed

in [54] for curve-grid intersection. We implemented a robust method, which was a modified version of the one presented in [54]. In this method, we checked four neighbors (E, W, N, S) of a given central pixel and found 16 combinations where the given contour could intersect the grid. Since the central pixel could be inside or outside, there were 16 positive combinations and 16 negative combinations. At the end of this process, we noted the distances of all the grid points which were closest to the given curve.

2. Tagging Stage:
   Here, we created three sets of grid points: The Accepted set included those points which lay on the given curve. All these points obviously had a distance of zero. We tagged them as ACCEPTED. If the curve did not pass through the grid points, then those points were the points of the initialization stage and we also tagged them as ACCEPTED. The Trial set included all points that were nearest neighbors to a point in the Accepted set. We tagged them as TRIAL. We then computed their distance values by solving the Eikonal Equation through expression (8.10). These points and their distances were put on the heap. The Far set were the grid points which were neither tagged as ACCEPTED nor TRIAL. We tagged them as FAR. They did not affect the distance computation of trial grid points. These grid points were not put onto the heap.

3. Marching Stage:

   (a) We first popped out a grid point (say P) from the top of the heap. It should have the smallest distance value among all the grid points in the heap. We tagged this point as ACCEPTED so that its value would not change anymore. We used the heap sort methodology for bubbling the least distance value on the heap.

   (b) We found the four nearest neighbors of the popped point P. This is what was done for these four points: if its tag was ACCEPTED, we did nothing; otherwise, we re-computed its distance by solving Eikonal equation (8.10). If it was FAR, it was relabeled as TRIAL and was put on the heap. If it is already labeled as TRIAL, its value was updated in the heap. This prevented the same point from appearing twice in the heap.

(c) Return to step (a) until there were no more points in the heap, i.e. all points had been tagged as ACCEPTED.

The superiority of this method is evidenced by the fact that we visited every grid point no more than four times. Here, the crux of the speed was due to the sorting algorithm. The grid or pixel location (x, y) was picked up by designing the back pointer method as used by Sethian et al. [53]. We used the heap sorting algorithm to select the smallest value (see Sedwick et al. [55]). Briefly, a heap can be viewed as a tree or a corresponding ordered array. A binary heap had the property that the value at a given "child" position int(i) is always larger than or equal to the value at its "parent" position (int (i/2)). The minimum travel time in the heap is stored at the top of the heap. Arranging the tentative travel time array onto a heap effectively identified and selected the minimum travel time in the array. The minimum travel time on the heap identified a corresponding minimum travel time grid point. Values could be added or removed from the heap. Adding or removing a value to/from the heap included re-arranging the array so that it satisfied the heap condition ("heapifying the array"). Heapifying an array was achieved by recursively exchanging the positions of any parent-child pair violating the heap property until the heap property was satisfied across the heap. Adding or removing a value from a heap generally has a computation cost of order O(**log** n), where n is the number of heap elements.

8.3 SVM for Level Set Initialization

The main issue in the application of SVM for snake initialization, as well as any other supervised statistical learning technique, is the training stage. During the training stage, first, representative images are segmented for region detection. Then, some feature extraction procedure must be applied, may be followed by principal component analysis (PCA) for dimensionality reduction [113, 114]. The results are used to train the SVM classifier (see [101] for training details).

In the segmentation task, a new image will be first classified by the trained SVM. The classifier must provide the initial contours which must be close to the correct boundaries. These contours are used to initialize the snake model in order to complete the boundary extraction (segmentation) process for the new image.

Such approach is applied in [98], in a segmentation framework for Computer Aided Dental X-rays Analysis. In this framework, authors apply a variational formulation of level set (see [98] and references therein). This approach combines energy minimization with the level set method. Therefore, it is suitable to segment an image that is modeled by energy functions, which is the approach followed in [98]. The feature used in this reference is a window-based feature processed by PCA. These features are stacked in feature vectors:

$$\mathbf{x}^i = \left(f_1^i, f_2^i, , ..., f_n^i\right)^T, \quad i = 1, 2, ..., s, \tag{8.11}$$

where f_j^i refers to the feature j for the image i.

Principal component analysis [113, 114], is applied to the feature vectors by computing the mean shape:

$$\overline{\mathbf{x}} = \frac{1}{s}\sum_{i=1}^{s}\mathbf{x}^i, \tag{8.12}$$

the covariance matrix:

$$\mathbf{S} = \frac{1}{s-1}\sum_{i=1}^{s}\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)\left(\mathbf{x}^i - \overline{\mathbf{x}}\right)^T, \tag{8.13}$$

and the eigenvectors of the covariance matrix:

$$\left\{\phi_1, \phi_2, ..., \phi_n\right\}. \tag{8.14}$$

The eigenvectors corresponding to the largest $t \leq n$ eigenvalues λ_i are retained in a matrix Φ. A vector $\mathbf{x}$ can now be approximated by:

$$\mathbf{x} = \overline{\mathbf{x}} + \Phi \cdot \mathbf{b}, \tag{8.15}$$

where $\mathbf{b}$ is a vector of t elements containing the model parameters, computed by:

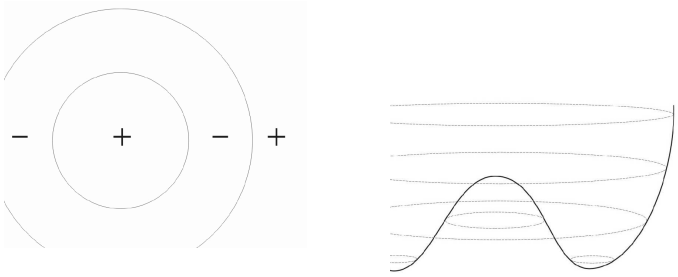
$$\mathbf{b} = \Phi^T \cdot \left(\mathbf{x} - \overline{\mathbf{x}}\right). \tag{8.16}$$

The corresponding $\mathbf{b}$ vectors will compose the feature space used for SVM training in [98]. Therefore, given a new X-ray image I, we can summarize the work presented in [98] as follows: (1) Pre-segmentation by level set in order to generate the training set; (2) Feature extraction to compute the $\mathbf{b}$ vector; (3) SVM

classification to provide the initial contours; (4) Variational level set initialization and final segmentation.

8.4 Dual-Level-Set Approach

In this section we maintain the philosophy of Dual-T-Snakes: one snake contracts and splits from outside the targets and another ones expand from inside the targets (section 6.3). However, the snake model will be an implicit one. To set ideas, let us consider the Figure 8.1.a, which shows two contours bounding the search space and Figure 8.1.b that pictures a surface which zero level set is the union of the two contours just presented.



(a) (b)

Figure 8.1: (a)Dual snakes bounding the search space. (b) Initial function which zero level set is the two contours presented.

If the surface evolves such that the two contours get closer, we can obtain the same behavior of Dual-T-Snakes. That is the key idea of the method proposed by us in [23]. In order to accomplish this goal we must define a suitable speed function and an efficient numerical approach. For simplicity, we consider the one dimensional version of the problem pictured on Figure 8.2. In this case, the level set equation given by expression (8.5) can be written as:

$$G_t + \frac{\partial G}{\partial x} F = 0. \tag{8.17}$$

The main point is to design the speed function F such that $G_t > 0$. Therefore, if we set the sign of F opposite to the one of G_x we get this goal, once:

$$G_t = -\frac{\partial G}{\partial x} F. \tag{8.18}$$

Hence, the desired behavior can be obtained by the sign distribution of F, shown in Figure 8.2.

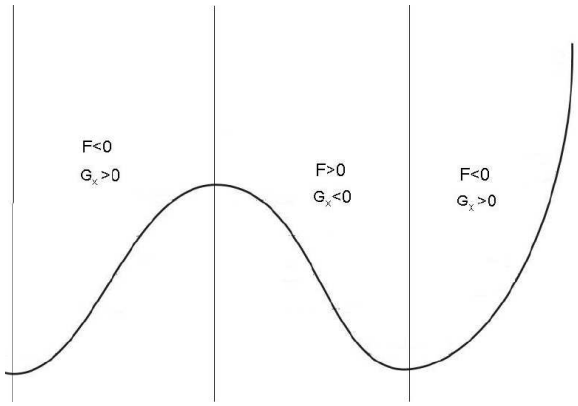


Figure 8.2: Sign of speed function.

However, we should notice that $G_x = 0$ for singular points. So, the values of G remain constant over these points because G_t becomes null. Thus, we should be careful about the surface evolution nearby the singular points because anomalies may happen. One possibility to avoid this problem is to stop front evolution before getting close to this point. Another possibility could be to change the evolution equation in order to allow the $G_t \neq 0$ over singular points. Such proposal implies that the isolines may be not preserved, that is, they become a function of time also. Thus:

$$G\left(x\left(t\right), t\right) = y\left(t\right), \tag{8.19}$$

consequently, by applying Chain rule:

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} = \frac{dy}{dt}. \tag{8.20}$$

Therefore, we should provide an speed function in y direction.

$$G_t + \left(\frac{\partial G}{\partial x}, -1\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}\right) = 0. \tag{8.21}$$

We can write this expression as:

$$G_t + F\,|(\nabla G, -1)| = 0, \tag{8.22}$$

where F is the speed function. For fronts in $3D$ we get:

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} + \frac{\partial G}{\partial y}\frac{dy}{dt} = \frac{dz}{dt}, \tag{8.23}$$

therefore:

$$G_t + \left(\frac{\partial G}{\partial x}, \frac{\partial G}{\partial y}, -1\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}, \frac{dz}{dt}\right) = 0. \tag{8.24}$$

One way to deal with these models is through viscous conservation laws [49]. For example, expression (8.20) becomes:

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} = \varepsilon\frac{\partial^2 G}{\partial x^2}, \tag{8.25}$$

if dy/dt is replaced by εG_{xx}, where ε is a new parameter. For $2D$ we will have:

$$G_t + \left(\frac{\partial G}{\partial x}, \frac{\partial G}{\partial y}\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}\right) = \varepsilon\nabla^2 G, \tag{8.26}$$

where ∇^2 means the Laplace operator defined by:

$$\nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}. \tag{8.27}$$

In our model we will maintain the idea that the front evolves in the normal direction. Thus, expression (8.26) can be rewritten as:

$$G_t + F\,|\nabla G| = \varepsilon\nabla^2 G, \tag{8.28}$$

following the same development to derive expression (8.5). Such model has been studied in the context of front propagation in [64, 8, 49] and can be used to achieve the desired result.

Once our application focus is shape recovery in a image I, we must choose a suitable speed function F as well as a convenient stopping term S to be added to the right-hand side of equation (8.28). Among the possibilities [32], the following ones have been suitable for our Dual-Level-Set:

$$F = -\frac{1 + \alpha k}{1 + |\nabla I|^2}, \tag{8.29}$$

$$S = \beta \nabla P \cdot \nabla G, \tag{8.30}$$

where k is the curvature, defined by expression (8.7), α, β are scale parameters and $P = -\left|\nabla I\right|^2$, like in equation (3.8). Therefore, we are going to deal with the following level set model:

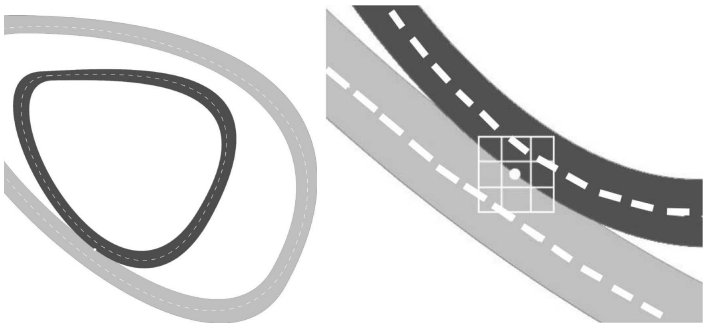
$$G_t = \left(\frac{1 + \alpha k}{1 + \left|\nabla I\right|^2}\right) \left|\nabla G\right| + \varepsilon \nabla^2 G + \beta \nabla P \cdot \nabla G. \tag{8.31}$$

The evolution of the fronts follows this governing equation and are interdependent due to the embedding function. However, once the evolution stops, we must evaluate the similarity between the two contours and apply a driving velocity instead of the driving force of section 5.4. The numerical method is a first order one already known in the level set literature [49]. We have also simplified the initialization of the method through smoothed versions of step functions (see [23], for details).

As usual in level set approaches, we use the narrow band method; that is, only the values of G within a tube placed around the front are updated. This approach can drop the complexity of the algorithm from $O\left(N^3\right)$, in three dimensions, to $O\left(mN^2\right)$, where m is the number of cells in the width of the narrow band [49]. When the front moves near to the edge of the tube boundary, the computation is stopped and a new tube is built with the zero level set at the center. For our dual approach, the narrow band is attractive not only for computational aspects but also because it allows an efficient way to evaluate similarity between two contours. In fact, instead of using the criterium of section 6.3, we take the procedure pictured on Figure 8.3: Firstly, the intersection point is computed (Figure 8.3.a); then, we take a neighborhood of this point (Figure 8.3.b) and stop to update the function G in all the grid points inside it or we can set to zero the speed function for these points. We say that those grid points are frozen ones.

Once the fronts stop moving, we must decide in which grid points we add a driving velocity. It is an extra velocity term which goal is the same of the driving force in section 6.3; that is, to keep fronts moving again. Therefore, we get a less sensitive model to the initial position of the fronts. To accomplish this task we can add an extra velocity term to equation (8.31), called V_{driv}.

We must be careful when choosing the grid points to apply this term. As in the case of Dual-T-Snakes, the fronts may be nearby the boundary somewhere,



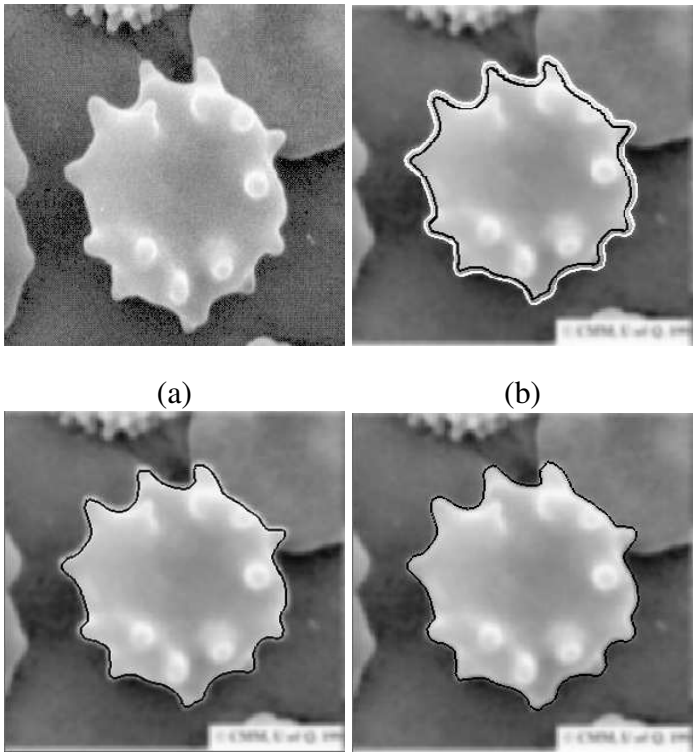
(a) (b)

Figure 8.3: (a)Narrow bands touching each other. (b) Neighborhood to define similarity between fronts.

but far away from the target in another place. We should automatically realize this fact when the fronts stop moving. To accomplish this, we can use the affinity operator explained on section 6.3. Based on this operator, we can define an affinity function that assigns to a grid point inside the narrow band a $0-1$ value: 0 for the grid points most likely to lie away from the target boundaries and 1 otherwise. Like in the Dual-T-Snakes, such affinity operator can be defined through fuzzy segmentation methods [65, 66, 67], image transforms [68, 69], region statistics [32], etc. The whole Dual-Level-Set algorithm can be summarized as follows: (1) Initialization through Step Functions; (2) Evolution until fronts stop. (3) Evaluate similarity. If frozen, stop. (4) Add V_{drive} for some time steps. (5) After that, turn-off V_{drive}. Go to step 2.

Figure 8.4 shows the application of the Dual-Level-Set to segment the cell image of Figure 8.4.a. However, instead of taking a band-pass version of it, like in Figure 6.2.b, we just apply a low pass filter to smooth the image. The Dual-Level-Set parameters are: $\alpha = 0.1, \varepsilon = 2.0, \beta = 0.1, T = 150, \Delta t = 0.05$.

The Figure 8.4.c shows the initialization for the greedy algorithm (section 5.3), and the Figure 8.4.d pictures the final result.



(a) (b)

(c) (d)

Figure 8.4: (a) Original image. (b) Dual-Level-Set result. (c)Initialization of the greedy snake model. (d) Final result.

8.5 Coupled Snakes

In [32] it is presented an approach that can be considered as an implicit formulation for twin snakes (section 5.6). In this method we have two coupled constrained fronts that evolve independently. Such method have been used in medical imaging applications where a volume has three tissue types, say $T1$, $T2$, and $T3$, and say tissue $T2$ was embedded in between tissues $T1$ and $T3$. Such an example is seen in the human brain where the GM (gray matter) is embedded between the WM (white matter) and CSF. The key point is that there is a coupling, a thickness constraint, between WM-GM and GM-CSF volumes. In fact, the cortical mantle has nearly constant thickness [71, 72]. Henceforth, Zengs proposed a level set method that starts with two embedded surfaces in the form of concentric sphere sets. The inner and outer surfaces were then evolved, driven by their own image-derived information, while maintaining the coupling in between through a thickness constraint. The model evolution is based on the system:

$$\frac{\partial G_{in}}{\partial t} + F_{in} \left|\nabla G_{in}\right| = 0, \tag{8.32}$$

$$\frac{\partial G_{out}}{\partial t} + F_{out} \left|\nabla G_{out}\right| = 0, \tag{8.33}$$

where G_{in} and G_{out} are the embedding functions of the inner and outer surface, respectively. In this model, the coupling between the two surfaces is obtained through the speed terms F_{in}, F_{out}, based on the distance between the two surfaces and the propagation forces. Therefore, we have two fronts which propagations are coupled only due to the distance between them. Therefore, it can be considered as an implicit formulation for twin snakes.

Chapter 9

Deformable Surface Approaches

9.1 Introduction

Snake models can be extended for surface reconstruction in $3D$ images generating deformable surface approaches [74]. The taxonomy pictured on Figure 3.2 can be applied also for deformable surface models.

Topologically adaptable surfaces are mainly represented by the T-Surfaces and Level Set [75, 8]. T-Surfaces model is the $3D$ counterpart of the T-Snakes, described on section 6.2. Level Set surfaces follow the formulation presented on section 8.2.

Dual surfaces follow the same philosophy of dual snake models: seek the global minimum through two deformable surfaces. Few works have been reported in the literature about dual surfaces[29, 29, 30, 37]. Although both the Dual-T-Snakes and Dual-Level-Set methods can be extended to $3D$, the former through the T-Surfaces model [14, 75] and the later through implicit formulations for deformable surfaces [49, 32, 76] , such implementations have not been reported in the literature yet.

Next, we summarize the T-Surfaces model its application for artery segmentation. Then, in section 9.3 we survey works in dual surfaces.

9.2 T-Surfaces

The T-Surfaces approach follows the same philosophy of its $2D$ version, described on section 6.2. It is composed by three components [14]: (a) a tetrahedral decomposition (CF-Triangulation) of the image domain $D \subset \Re^3$; (b) a particle model of the deformable surface; (c) a *Characteristic Function* χ defined on the grid nodes which distinguishes the interior ($Int(S)$) from the exterior ($Ext(S)$) of a surface S:

$$\chi : D \subset \Re^3 \rightarrow \{0, 1\} \tag{9.1}$$

where $\chi(p) = 1$ if $p \in Int(S)$ and $\chi(p) = 0$, otherwise (p is a node of the grid).

Following the classical nomenclature [89], a tetrahedron (also called a simplex) σ is a *transverse* one if the characteristic function χ in equation (9.1) changes its value in σ. Analogously, for an edge.

In the framework composed by both the simplicial decomposition and the characteristic function, the reparameterization of a surface is done by [82, 14]: (1) Computing the intersections of the surface with the grid; (2) Finding the set of transverse tetrahedra (*Combinatorial Manifold*); (3) Choosing a surface point for each transverse edge; (4) Connecting the selected points.

In this reparameterization process, the transverse simplices play a central role. Given such a simplex, we choose in each transverse edge an intersection point to generate the new surface patch. In general, we will obtain three or four transverse edges in each transverse tetrahedron (Figure 9.1). The former gives a triangular patch and the later defines two triangles. So, at the end of the step (4), a triangular mesh is obtained and topological changes are resolved likewise in the $2D$ case, described on section 6.2. Each triangle is called a *triangular element* [14].

Figure 9.1: Basic types of intersections between a plane and a simplex in 3D.

After the reparameterization process, a suitable evolution scheme must be ap-

plied. Dynamically, a T-Surfaces can be seen as a closed elastic mesh [14]. Each mesh node is called a *node element* and each pair of connected nodes v_i, v_j is called a *model element*.

The node elements are linked by springs, whose natural length we set to zero. Hence, a tensile force can be defined by:

$$\overrightarrow{\alpha_i} = \sum_j \overrightarrow{S}_{ij} \text{ where } \overrightarrow{S}_{ij} = c \cdot r_{ij}, \tag{9.2}$$

c is a scale factor and $r_{ij} = \|v_i - v_j\|$ is the length of the corresponding model element. The model also has a normal force which can be weighted as follows [14]:

$$F_i = k\left(sign_i\right) n_i, \tag{9.3}$$

where n_i is the normal vector at node i, k is a scale factor, and $sign_i = +1$ if $I\left(v_i\right) > T$ and $sign_i = -1$ otherwise (T is a threshold of the image I). This force is used to push the model towards image edges until it is opposed by external image forces.

The forces defined by the equations (9.2)-(9.3) are internal forces. The external force is defined as a function of the image data, according to the interested features. Several different approaches have been adopted according to the application [42, 94]. In our case, it can be defined as follows:

$$Image :: Force :: f_i^t = -\gamma_i \nabla P; \quad P = \|\nabla I\|^2. \tag{9.4}$$

The evolution of the surface is controlled by the following dynamical system:

$$v_i^{(t+\Delta t)} = v_i^t + h_i \left(\overrightarrow{\alpha_i}^t + \overrightarrow{F_i}^t + \overrightarrow{f_i}^t\right), \tag{9.5}$$

where h_i is an evolution step.

During the T-Surfaces evolution, some grid nodes become interior to a surface. Such nodes are called *burnt nodes* and its identification is required by the update of the characteristic function [14]. To deal with self-intersections, the T-Surfaces model incorporates an entropy condition: *once a node is burnt it stays burnt*. A termination condition is set based on the number of deformation steps in which a simplex has remained a transverse one (*freezing point*). After the identification of burnt nodes, we apply steps (1)-(4) for reparameterization and topological changes.

9.2.1 Artery Reconstruction

This section demonstrates the advantages of applying T-Surfaces plus isosurface methods. Firstly, we segment an artery from an $80 \times 86 \times 72$ image volume obtained from the Visible Human project. This in an interesting example because the intensity pattern inside the artery is not homogeneous.

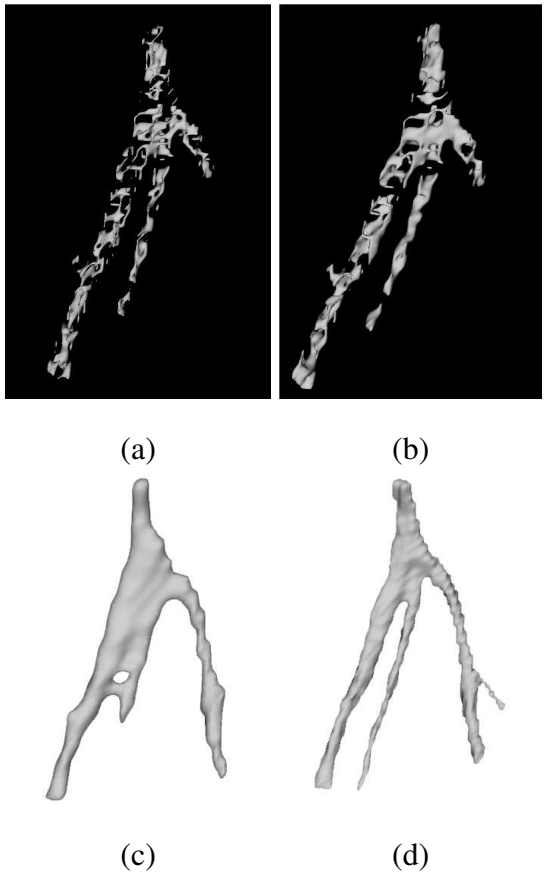


Figure 9.2: (a) Initialization with grid $3 \times 3 \times 3$. (b)T-Surfaces evolution (step 1). (c)Solution for initial grid. (d)Final solution for grid $1 \times 1 \times 1$.

Figure 9.2.a shows the initialization when using the intensity range $(28, 32)$ to initialize the characteristic function (equation (9.1)); that means:

$$\begin{aligned} \chi(p) = 1 \quad if \quad I(p) \in (28, 32), \\ \chi(p) = 0, \quad otherwise, \end{aligned} \tag{9.6}$$

where $I(p)$ is the image intensity at the node p of the grid. The extracted topology is too different from that one of the target. However, when applying T-Surfaces the obtained geometry is improved. Figure 9.2.b shows the result after the first step of evolution. The merges among components improve the result. After 4 interactions of the T-Surfaces algorithm, the extracted geometry becomes closer to that one of the target (Figure 9.2.c).

However, the topology remains different. The problem in this case is that the used grid resolution is too coarse if compared with the separation between branches of the structure. Thus, the flexibility of the model was not enough to correctly perform the surface reconstruction.

The solution is to increase the resolution and to take the partial result of Figure 9.2.c to initialize the model in the finer resolution. In this case, the correct result is obtained only with the finest grid $(1 \times 1 \times 1)$. Figure 9.2.d shows the desired result obtained after 9 interactions. We also observe that new portions of the branches were reconstructed due to the increasing of T-Surfaces flexibility obtained through the finer grid. We should emphasize that an advantage of the multiresolution approach is that at the lower resolution, small background artifacts become less significant relative to the object(s) of interest. Besides, it avoids the computational cost of using a finer grid resolution to get closer the target.

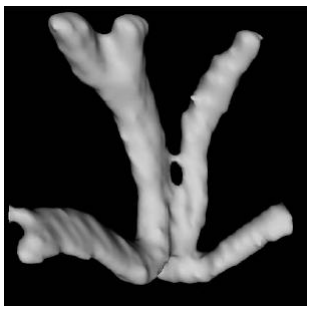
The T-Surfaces parameters are $\gamma = 0.01$, $k = 1.222$; $c = 0.750$. The total number of evolution is 13. The number of triangular elements is 10104 for the highest resolution and the clock time was of order of 3 minutes.

Sometimes, even the finest resolution may not be enough to get the correct result. Figure 9.3.a pictures such an example.

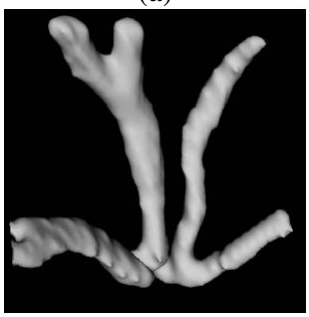
In this case, we segment an artery from an $155 \times 170 \times 165$ image volume obtained from the Visible Man Project. The T-Surfaces parameters are: $c = 0.75$, $k = 1.12$, $\gamma = 0.3$, grid resolution is $4 \times 4 \times 4$ and freezing point is set to 10. The result of T-Surfaces evolution is pictured on Figure 9.3.a. Likewise in the last example, the initialization was performed by computing the characteristic function in the $4 \times 4 \times 4$ grid, using expression (9.6) .

Among the proposal to address this problem (relax the threshold, mathematical morphology [115], etc), we tested the anisotropic diffusion [93]. The properties of this method (section 4.2) enable smoothing within regions in preference to smoothing across boundaries. Figure 9.3.b shows the correct result obtained when

pre-processing the image with anisotropic diffusion.



(a)



(b)

Figure 9.3: (a)Example showing an incorrect result. (b)Final result improved by pre-processing with anisotropic diffusion.

9.3 Dual Surfaces Methods

The formulation of the dual-front approach, described on section 7.5, remains the same for $2D$ and $3D$ and the fast marching algorithm used to solve expressions (7.40)-(7.42) is also efficient for $3D$. In [28] it is proposed a $3D$ dual-front approach for brain cortex segmentation that allows user interact with a partial result to improve it. In this scheme the search space is defined by the image histogram analysis to find out the voxels between the CSF and WM tissues and voxels between the CSF and WM tissues. The user interaction is performed by just adding useful seed points with simple mouse clicks.

In [29, 31] a parametric dual surface model is presented for automatic segmentation of structures from noisy PET images. The deformable model is based on an greedy algorithm for the energy minimization . In this approach, the surface is represented by a deformable mesh $S = \{v_1, v_1, .., v_N\} \subset \Re^3$. Adjacency relations are known and constant, hence symbol S is used to name the corresponding polygonal mesh. The total energy of the surface mesh S is like expression (5.35):

$$E(S) = \lambda E_{int}(S) + (1-\lambda) E_{ext}(S), \tag{9.7}$$

$$= \frac{1}{N} \sum_{i=1}^{N} (\lambda E_{int}(v_i) + (1-\lambda) E_{ext}(v_i)). \tag{9.8}$$

The parameter λ is in range $[0, 1]$ and it weigh the contribution of the internal (E_{int}) and external (E_{ext}) energy of the model. Like in the snake approaches, the external energy couples the model to the target image features and the internal energy controls the smooth of the deformable surface. They are defined as follows:

$$E_{int}(v_i) = \frac{1}{A(S)} \left\| v_i - \alpha \sum_{j=1}^{3} v_{ij} \right\|^2, \tag{9.9}$$

where v_{ij} are the neighboring points of v_i in the mesh, $A(S)$ is the average area of the faces of the surface, and α is a shape parameter. In the simplest case, the thin-plate shape model, it is set $\alpha = 1/3$. Other models shape model are considered in [30].

The external energy is defined as:

$$E_{ext}(v_i) = 1 - \frac{\|\nabla I(v_i)\|}{\max_{x \in D} \|\nabla I(x)\|}, \tag{9.10}$$

where D is the image domain. This energy is a monotonically decreasing function of the gradient intensity. If compared with expression (5.34) we observe that it has the advantage of been less sensitive to weak edges due to the normalization used.

Like in the snake models (section 3), in general the energy function defined by expression (9.7) is not convex. In [29] it is proposed a dual surface minimization (DSM) algorithm for the global optimization task. The algorithm is based on the iterative minimization of the energies of two surfaces defined by an outer mesh and the inner mesh. The energies for the both surfaces are calculated from equations

(9.9)-(9.10). During each iteration, the energy of the mesh is minimized using a greedy method adapted from [77] (see also section 5.3). The energy minimization is used in such a way, that the outer surface shrinks and the inner surface grows. Following the dual snake algorithm of section 5.4, after each iteration, the DSM algorithm compares the energies of the outer and inner surfaces and continues minimization with the surface having higher energy. If the surface having higher energy gets stuck in a local minimum, the energy of the current surface position is increased until the surface moves again. The DSM algorithm is stopped when the volume inside of the inner surface exceeds the volume of the outer surface. The algorithm selects the surface having the lower energy as the result.

The DSM algorithm has been applied for surface reconstruction in PET images. The use of PET scanners [78, 79] has increased which creates a need for automatic methods for analysis of this kind of data. A PET image of the brain can be registered (rigidly) with the MR images of the same structure to allow gathering of physiological information associated with the anatomical structures. However, PET images are noisy which increases the difficulties for automatic surface extraction. That is the motivation for applying dual surfaces to segment PET data.

In [29] it is presented a study of the DSM algorithm for extraction of brain surfaces from $3D$ PET images. The surface reconstruction was accomplished by using the deformable mesh with the internal energy given by expression (9.9), with $\alpha = 1/3$, and the external energy calculated from expression (9.10). The initialization for DSM was performed by two concentric ellipsoids, as pictured in Figure 9.4.

A visual inspection indicates that the extracted brain surfaces were accurate in all the 17 studied cases, as we can check in Figure 9.5 which pictures a typical example.

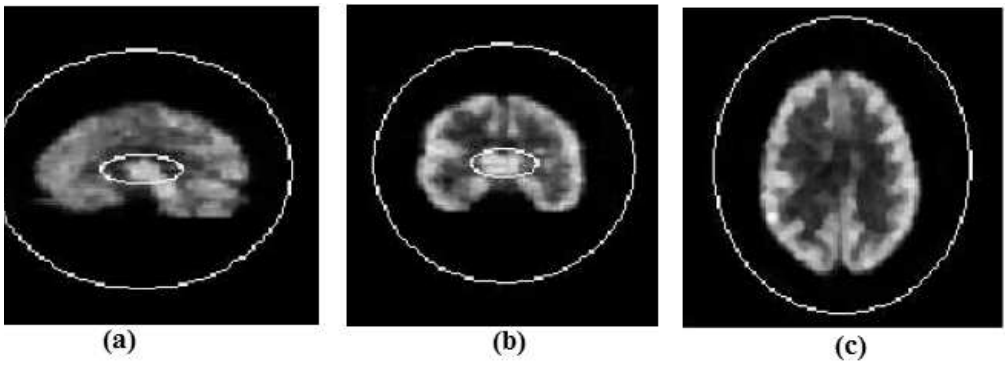


Figure 9.4: Initialization of the DSM: (a) Sagittal view. (b) Coronal view. (c) Transaxial cross-section view.

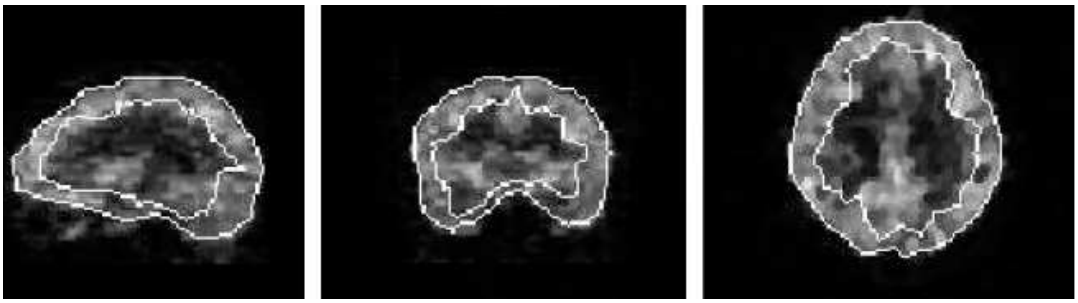


Figure 9.5: Three cross-section views of a typical result obtained for brain surfaces reconstruction from $3D$ PET images. (a) Sagittal view. (b) Coronal view. (c) Transaxial cross-section view.

Chapter 10

Discussion

The non-invariance under affine transformations implies that the internal energy is sensitive to distortions due to changes in viewing geometry. It means that the elastic forces may affect the efficiency of the energy minimization process [19, 20]. The AI-Snake - a discrete and affine-invariant snake model [19] - addressed this problem. However, we pointed out also the need for invariance against reparameterization also.

For affine-invariant snakes, a complete description of the reparameterization problem comes from the Lie Theory. In this case, we have to account for two (Lie) groups: the reparameterization and affine transformations. Although there is no a common invariant for both groups the results presented on section 5.2.1, towards the exploration of integral invariants, shall be considered in the context of Lie Groups for snake models.

Hierarchical filtering methods, as well as a continuation method based on a (discrete) linear scale-space representations is a general approach to reduce the nonconvexity problems. The idea described in the Figure 7.2 is robust and can be applied to deal with noisy and textured images.

In deformable models area, defining the initial estimation from that we can start the model evolution (the initialization step) is a difficult and important task. Problems associated with fitting the model to data could be reduced if a better start point for the search were available. The application of supervised statistical learning methods (sections 6.4 and 8.3) is an interesting area. However, the training stage is a drawback which limits the range of applications.

An important point for dual snakes is how to proceed the evolution after both snakes comes at rest but are far from each other. With exception of the cell-based dual model, the original dual model (sections 5.7 and 5.4) and the Dual-Front (section 7.5), all the presented dual approaches apply some procedure in this step, in order to avoid that some snaxels pass over the desired boundary, due to the driving term. We can say that the notion of stability of the Dual-DGF, defined in section 5.5, is the counterpart of the affinity operator in the Dual-T-Snakes and Dual-Level-Set models.

Such procedure is bypassed in the original dual model (section 5.4) due to the shape model used. The corresponding energy term may prevents that a snake pass over a global minimum. However, the shape model given by expression (5.30) limits the application of the method for general shapes and topologies. On the other hand, in the cell-based dual model, the cell decomposition of the image domain provided by the watershed strongly reduces the search space. Therefore, the energy minimization method may be less sensitive to local minima. Probably, that is way authors do not take care about the possibility of snakes reach equilibrium position far from the target. However, it does not seems to be possible to prove that such situation never happens. In the Dual-Front, actually the level sets of the action map U give the evolution of the front. The velocity of the evolving front is decided by the potential. Therefore, both the potentials in expressions (7.43)-(7.44) must be defined such that the velocity is much lower when the evolving fronts achieve the boundary. If the constant $w > 0$, the front velocity will be never null, and so, fronts only stop when the two action maps meet each other. Such policy is interesting to pass over local minima but once there is no an energy balance between fronts the global minimum may be also lost. Even the user interaction proposed in [28] is not enough to address this problem because fronts may met too far way from the target, mainly if w value is not properly choose.

Dual-Level-Set as well as Dual-T-Snakes are topologically adaptable deformable models which increases their range of applications. However, such generality has also a price: the care with local minima should be higher than for example, in the original model, because there is no a shape model to bias the solution to the desired shape.

Despite of the capabilities to reject local minima, dual models have also some disadvantages. Firstly, the method is at least two times more expensive than single

approaches. Secondly, the initialization may be a tedious task because the user should set two curves at the beginning of the process. The choice of parameter values is also another point to be careful because in this case there are two snakes to be set.

10.1 Conclusions and Future Works

Despite of their abilities to integrate image data and desired contour properties, traditional parametric snake models have limitations due to the non-invariance of the internal energy under affine transformations, the non-convexity of the model energy functional and the inability to deal with topological changes.

The problem of topological changes has been addressed by embedding the snake in the framework of a simplicial decomposition of the domain or through implicit formulations.

The original dual approach is an interesting technique to address the sensitivity to local minima of usual snake models. The idea of using two snakes to seek for the global minimum, originally proposed in [17], have been used and extended in recent works. Topological capabilities were incorporated, implicity formulations were developed and a cell-based approach as well as new schemes to set the image force and the snake evolution were designed in order to improve the efficiency.

Lie Group Theory, a natural and complete theoretical tool for invariance, could be worthwhile for computer vision researchers, not only due to its application in active shape models but also in other areas like curve matching, planar shape recognition, constrained stereo and invariant moments theory.

The non-convexity of the model energy can be also addressed through an automatic procedure to initialize the model closer to the desired boundary. The automatic initialization of snakes by supervised statistical learning methods is an interesting problem that must be considered more deeply in the near future. It provides also a powerful methodology to incorporate prior knowledge of shape and image pattern into the segmentation. Besides, the implementation of Dual-T-Snakes and Dual-Level-Set for $3D$ shall be considered in order to get dual and topologically adaptable deformable surface models.

Bibliography

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

[2] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: A survey. *Medical Image Analysis*, 1(2), 1996.

[3] 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.

[4] B.M. ter Haar Romery W.J. Niessen and M.A. Viergever. Geodesic deformable models for medical image analysis. *IEEE Trans. on Medical Imaging*, 17(4), August 1998.

[5] G. Sapiro. Color snakes. *Computer Vision and Image Understanding*, 68(2):247–253, 1997.

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

[7] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[8] 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.

[9] Jasjit S. Suri, Kecheng Liu, Sameer Singh, Swamy Laxminarayan, Xiaolan Zeng, and Laura Reden. Shape recovery algorithms using level sets in 2-d/3-d medical imagery: a state-of-the-art review. *IEEE Transactions on Information Technology in Biomedicine*, 6(1):8–28, 2002.

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

[11] 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.

[12] Antonio A. F. Oliveira, Saulo Ribeiro, Ricardo C. Farias, Claudio Esperança, and Gilson A. Giraldi. Loop snakes: Snakes with enhanced topology control. In *SIBGRAPI*, pages 364–371, 2004.

[13] S. Bischoff and L. Kobbeit. Snakes with topology control. *The Visual Computer*, 2004.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] G. A. Giraldi, N. Vasconcelos, E.Strauss, and A. F. Oliveira. Dual and topologically adaptable snakes and initialization of deformable models. Technical report, National Laboratory for Scientific Computation, Brazil, http://www.lncc.br/proj-pesq/relpesq-01.htm, 2001.

[19] H H. S. Ip and D. Shen. An affine-invariant active contour model (ai-snake) for model-based segmentation. *Image and Vision Computing*, 16(2):135–146, February 1998.

[20] M. A. Snyder. On the mathematical foundations of smoothness constraints for the determination of optical flow and for surface reconstruction. *IEEE Trans. on Pattern Analusis and Mach. Intell.*, 13(11):1105–1114, November 1991.

[21] G. Giraldi and A.A.F.Oliveira. Invariant snakes and initialization of deformable models. *International Journal of Image and Graphics*, 4(3), Jully 2004.

[22] 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.

[23] Jasjit Suri and Aly Farag (Editors), editors. *deformable models: clinical and biological applications*. Springer, NY, 2006.

[24] Hua Li and A. Yezzi. Local or global minima: Flexible dual-front active contours. In *Lecture Notes in Computer Science: Computer Vision for Biomedical Image Applications*, volume 3765, pages 356–366. Springer, Berlin, Heidelberg, 2005.

[25] L. Cohen. Multiple contour finding and perceptual grouping using minimal paths. *Journal of Mathematical Imaging and Vision*, 14:225–236, 2001.

[26] L. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. In *IEEE International Conference on CVPR (CVPR'96)*, pages 666–673, 1996.

[27] J. A. Sethian. Fast marching methods. *SIAM Review*, 41:199–235, 1999.

[28] Hua Li, Anthony J. Yezzi, and Laurent D. Cohen. Fast 3d brain segmentation using dual-front active contours with optional user-interaction. In *CVBIA*, pages 335–345, http://dx.doi.org/10.1007/11569541_34, 2005.

[29] J. Mykkänen, J. Tohka, and U. Ruotsalainen. Delineation of brain structures from positron emission tomography images with deformable models. In *In The New Navigators: From Professionals to Patients (Proc. MIE03)*, pages 33–38, 2003.

[30] J. Tohka and J. Mykkänen. Deformable mesh for automated surface extraction from noisy images. *Int. J. Image and Graphics*, 4(3):405432, 2004.

[31] J. Mykkänen, J. Tohka, J. Luoma, and U. Ruotsalainen. Automatic extraction of brain surface and mid-sagittal plane from pet images applying deformable models. Technical report, Tech. Rep. Dept. Comput. Information Sciences, Univ. Tampere, Finland., www.cs.uta.fi/reports/r2003.html.

[32] Jasjit S. Suri, Kecheng Liu, Sameer Singh, Swamy Laxminarayan, Xiaolan Zeng, and Laura Reden. Shape recovery algorithms using level sets in 2-d/3-d medical imagery: a state-of-the-art review. *IEEE Transactions on Information Technology in Biomedicine*, 6(1):8–28, 2002.

[33] 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.

[34] DA-MING SHI, P.w.s. HENG, and FEI CHEN. A pellet sphericity measure system based on dual active contour models. In *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, November 2003.

[35] CHUNG-MING CHEN, HENRY HORNG-SHING LU, and AN-TING HSIAO. A dual-snake model of high penetrability for ultrasound image boundary extraction. *Ultrasound in Med. Biol.*, 27(12):1651-1665, 2001.

[36] CHUNG-MING CHEN, HENRY HORNG-SHING LU, and YUENG-SHIANG HUANG. Cell-based dual snake model: A new approach to extracting highly winding boundaries in the ultrasound images. *Ultrasound in Med. Biol.*, 28(8):1061-1073, 2002.

[37] J. Tohka, A. Kivimäki, A. Reilhac, J. Mykkänen, and U. Ruotsalainen. Assessment of brain surface extraction from pet images using monte carlo simulations. *IEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 51, NO. 5,*, 51(5):2641–2648, OCTOBER 2004.

[38] 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.

[39] Mohammad Dawood, Xiaoyi Jiang, and Klaus P. Schäfers. Reliable dual-band based contour detection: A double dynamic programming approach. In *ICIAR (2)*, pages 544–551, 2004.

[40] G. Georgoulas, G. Nikolakopoulos, Y. Koutroulis, A. Tzes, and P. Groumpos. An intelligent visual-based system for object inspection and welding, relying on active contour models-algorithms. In *Proceedings of the 2nd Hellenic Conference on AI*, volume Greece Companion Volume, pages 399–410, Thessaloniki, April 2002.

[41] G.B. Aboutanos, J. Nikanne, N. Watkins, and B.M. Dawant. Model creation and deformation for the automatic segmentation of the brain in mr images. *IEEE Transactions on Biomedical Engineering*, 46(11):1346–1356, 199.

[42] G. A. Giraldi, E. Strauss, and A. F. Oliveira. Dual-T-Snakes model for medical imaging segmentation. *Pattern Recognition Letters*, 24(7):993–1003, 2003.

[43] L. Vincent and P. Soille. Watershed in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Trans PAMI*, 13(6):583-597, 1991.

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

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

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

[47] T. McInerney and D. Terzopoulos. T-snakes: Topology adaptive snakes. *Medical Image Analysis*, 4(2):73–91, June 2000.

[48] G. A. Giraldi, E. Strauss, and A. F. Oliveira. Improving the dual-t-snakes model. In *International Symposium on Computer Graphics, Image Processing and Vision (SIBGRAPI'2001), Florianópolis, Brazil,October 15-18*, pages 346–353, 2001.

[49] J. A. Sethian. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*. Cambridge University Press, 1996.

[50] J.A. Sethian. A fast marching level set method for monotonically advancing fronts. In *Proceedings of the National Academy of Sciences*, volume 93, pages 1591–1595, 1995.

[51] S. Cao and S. Greenhalgh. Finite-difference solution of the eikonal equation using an efficient, first-arrival, wavefront tracking scheme. *Geophysics*, 59(4):632–643, April 1994.

[52] S. Chen, B. Merriman, S. Osher, and P. Smereka. A simple level set method for solving stefan problems. *J. Comput. Phys.*, 135(1):8–29, 1997.

[53] J.A. Sethian. Three-dimensional seismic imaging of complex velocity structures. *US Patent*, Jan. 2000.

[54] D. Adalsteinsson and J. Sethian. The fast construction of extension velocities in level set methods. *Journal of Computational Physics*, 148:2–22, 1998.

[55] R. Sedgewick. *Algorithms in C, Fundamentals, data structures, sorting, searching*, volume 1. Addison-Wesley, 1998.

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

[57] C.S. Poon and M. Braun. Image segmentation by a deformable contour model incorporating region analysis. *Physics in Medicine and Biology*, 42:1833–1841, 1997.

[58] Donna J. Williams and Mubarak Shah. A fast algorithm for active contours and curvature estimation. *CVGIP: Image Underst.*, 55(1):14–26, 1992.

[59] P. Bamford and B. Lovell. Robust cell nucleous segmentation using a viterbi search based active contour. In *Proc. Of the Fifth Int. Conf. On Computer Vision (ICCV'95), Cambridge, MA, USA*, pages 840–845, June 1995.

[60] S.R. Gunn and M.S. Nixon. A dual active contour including parametric shape. Technical report, University of Southampton: Research Journal, http://www.ecs.soton.ac.uk/publications/rj/1994/vssp/gunn/steve.html, 1994.

[61] S.R. Gunn and M.S. Nixon. Snake head boundary extraction using global and local energy minimisation. In *Proc. IEEE Internat. Conf. on Pattern Recognition B*, pages 581–585, 1996.

[62] C.M. Chen, H.H.S Lu, and Y.C. Lin. An early vision based snake model for ultrasound image segmentation. *Ultrasound Med. and Bio.*, 26(2):273-285, 2000.

[63] Y.H. Lee and S.A. Kassman. Generalized median filtering and related nonlinear filtering techniques. *IEEE Trans Acoust Speech Signal Processing*, 33:672-683, 1985.

[64] S. Osher J. A. Sethian. Fronts propagation with curvature-dependent speed: Algorthms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

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

[66] J. C. Bezdek and L. O. Hall. Review of mr image segmentation techniques using pattern recognition. *Med. Phys.*, 20(4):1033-1048, March 1993.

[67] C. Xu, D. Pham, M. Rettmann, D. Yu, and J. Prince. Reconstruction of the human cerebral cortex from magnetic resonance images. *IEEE Trans. Med. Imag.*, 18(6):467–480, June 1999.

[68] R. Pohle, T. Behlau, and K. D. Toennies. Segmentation of 3-D medical image data sets with a combination of region based initial segmentation and active surfaces. In *Proceedings of SPIE, (Medical Imaging 2003)*, pages 69–76, February 2003.

[69] A.X. Falco, B.S. da Cunha, and R.A. Lotufo. Design of connected operators using the image foresting transform. In *SPIE on Medical Imaging*, volume 4322, pages 468–479, February 2001.

[70] Martin Kerschner. Homologous twin snakes integrated in a bundle block adjustment. Technical report, citeseer.ist.psu.edu/kerschner98homologous.html.

[71] X. Zeng, L. H. Staib, R. T. Schultz, and J. S. Duncan. Segmentation and measurement of the cortex from 3d mr images. In *Med. Image Comput. Computer-Assisted Intervention*, page 519-530, 1998.

[72] X. Zeng, L.H. Staib, R.T. Schultz, and J.S. Duncan. Segmentation and measurement of the cortex from 3-d mr images using coupled-surfaces propagation. *IEEE Transactions on Medical Imaging*, 18(10):927–937, 1999.

[73] Benjamin Hohnhäuser and Günter Hommel. 3d pose estimation using coupled snakes. In *WSCG*, pages 161–166, 2004.

[74] Dimitris N. Metaxas. *Physics-Based Deformable Models Applications to Computer Vision, Graphics and Medical Imaging*. Springer, 1997.

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

[76] Joshua E. Cates, Aaron E. Lefohn, and Ross T. Whitaker. GIST: An interactive, GPU-based level set segmentation tool for 3D medical images.

[77] D.J.Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *CVGIP: Image Understanding*, 55(1):14-26, 1992.

[78] M. Phelps and J. Mazziotta. Positron emission tomography: Human brain function and biochemistry. *Science*, 228(4701):799-809, 1985.

[79] P. Suetens. *Fundamentals of Medical Imaging*. Cambrigde University Press, New York., 2002.

[80] 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), 2000.

[81] 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.

[82] 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.

[83] V. Chalana, D. Haynor, P. Sampson, and Y. Kim. Parameter estimation in deformable models using Markov chain Monte Carlo. In *Proc. SPIE Vol. 3034, p. 287–298, Medical Imaging 1997: Image Processing, Kenneth M. Hanson; Ed.*, pages 287–298, April 1997.

[84] 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.

[85] P.S. Rodrigues and G.A. Giraldi. Parameter estimation with a bayesian network in medical image segmentation. In *Computer Graphics and Imaging (CGIM)*, Kauai, Hawaii, USA, August 2004.

[86] Douglas DeCarlo and Dimitris N. Metaxas. Adjusting shape parameters using model-based optical flow residuals. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(6):814–823, 2002.

[87] Q. Liang, I. Wendelhag, J. Wikstrand, and T. Gustavsson. A multiscale dynamic programming procedure for boundary detection in ultrasonic artery images. *IEEE Trans. on Medical Imaging*, 19(2):127–142, 2000.

[88] M. Mignotte and J. Meunier. An unsupervised multiscale approach for the dynamic contour-based boundary detection issue in ultrasound imagery. 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 366–369, February, 2000.

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

[90] 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.

[91] B. Lovell P. Bamford. Unsupervised cell nucleus segmentation with active contours. Technical report, Dep. Eletrical and Comp. Eng., Univ. of Queensland, Australia, http://www.cssip.elec.uq.edu.au/, 1998.

[92] B. A. Dubrovin, A. T. Fomenko, and S. P. Novikov. *Modern Geometry - Methods and Applications (The Geometry of Surfaces, Transformation Groups, and Fields*, volume 1. Springer-Verlag, 1984.

[93] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on Patter Analusis and Mach. Intell.*, 12(7):629–639, July 1990.

[94] C. Xu and J. Prince. Snakes, shapes, and gradient vector flow. *IEEE Trans. Image Proc.*, pages 359–369, March 1998.

[95] 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.

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

[97] K.W. Cheung, D.Y. Yeung, and R.T. Chin. On deformable models for visual pattern recognition. *PATTERN RECOGNITION*, 35:1507 –1526, 2002.

[98] Shuo Li, T. Fevens, A. Krzyzak, and Song Li. An automatic variational level set segmentation framework for computer aided dental x-rays analysis in clinical environments. *Computerized Medical Imaging and Graphics*, 30(2):65–74, 2006.

[99] C. Xu and J. L. Prince. Global optimality of gradient vector flow. In *Proc. of the Conference on Information Sciences and Systems, Princeton University*, March 2000.

[100] P. J. Olver. *Applications of Lie Groups to Differential Equations*. Springer-Verlag, Berlin, (1986), 1986.

[101] Vladimir N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, INC., 1998.

[102] J. J. Staal B. Haar Romeny M. A. Viergever B. van Ginneken, A. F. Frangi. A non-linear gray-level appearance model improves active shape model segmentation. In *IEEE Workshop on Mathematical Models in Biomedical Image Analysis, MMBIA 2001*, pages 205–212, 2001.

[103] Kok F. Lai and R.Chin. Deformable contour: modeling and extraction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(11):1084–1090, 1995.

[104] Jun Sato and R. Cipolla. Affine integral invariants for extracting symmetry axes. *Image and Vision Computing*, 15:627–635, 1997.

[105] Henry Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. In *Computer Vision and Pattern Recognition '96*, June 1996.

[106] S. J. Gilson, I. Middleton, and R. I. Damper. Neural techniques for outlining the lungs from mr images of the thorax. *In proceedings of Symposium of Softwere Computing World Automation Congress*, May 1998.

[107] Y. Chauvin and D. Rumelhart. *Backpropagation: Theory, Architectures and Applications*. Lawrence Erlbaum, Hillsdale, NJ, 1995.

[108] D. E. Rumelhart, G. Hinton, and R. Williams. Learning representations by back-propagation errors. *Nature*, (323):533–536, 1986.

[109] A. Weigend, D. Rumelhart, and B. Huberman. Generalization by weight-elimination with application to forecasting. *Advances in Neural Information Processing 3*, pages 875–882, 1991.

[110] A. Witkin. Scale-space filtering. In *Int. Joint Conf. Artificial Intelligence, Karlsruhe, Germany*, pages 1019–1021, 1983.

[111] J. Koenderink. The structure of images. *Biol. Cybern.*, 50:363–370, 1984.

[112] H. Goldstein. *Classical Mechanics*. Addison-Wesley, 2nd edition, 1981.

[113] T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[114] I. T. Jolliffe, B. J. T. Morgan, and P. J. Young. A simulation study of the use of principal component in linear discriminant analysis. *Journal of Statistical Computing*, 55:353–366, 1996.

[115] A. Sarti, C. Ortiz, S. Lockett, and R. Malladi. A unified geometric model for 3d confocal image analysis in cytology. In *Proc. International Symposium on Computer Graphics, Image Processing, and Vision (SIBGRAPI'98)*, pages 69–76, 1998.

[116] R. Beale and T. Jackson. *Neural Computing*. MIT Press, 1994.

[117] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[118] G. A. Giraldi and A. F. Oliveira. Convexity analysis of snake models based on hamiltonian formulation. Technical report, http://arxiv.org/abs/cs/0504031, 1999.

[119] Hong Qin and Demetri Terzopoulos. D-nurbs: A physics-based geometric design framework. *IEEE Transactions on Visualization and Computer Graphics,2(1*, 2:85–96, 1996.

[120] C. Hirsch. *Numerical Computation of Internal and External Flows*, volume 1. Livros Tcnicos e Cientficos Editora S.A., RJ-Brasil, 1988.

[121] C.A. Davatzikos and J.L. Prince. Convexity analysis of active contour algorithms. *Image and Vision Computing*, 17(1):27–36, January 1999.

[122] G. H. Golub and C. F. Van Loan. *Matrix Computation*. The Johns Hopkins University Press, 1985.