# THÈSE

présentée à

**L'UNIVERSITÉ de PARIS-DAUPHINE**

pour obtenir le grade de

**DOCTEUR EN SCIENCES**

Spécialité

**MATHÉMATIQUES APPLIQUÉES ET TRAITEMENT D'IMAGES**

par

**Thomas DESCHAMPS**
http://math.lbl.gov/ deschamp

Sujet de la thèse :

# EXTRACTION DE COURBES ET SURFACES PAR MÉTHODES DE CHEMINS MINIMAUX ET ENSEMBLES DE NIVEAUX. APPLICATIONS EN IMAGERIE MEDICALE 3D.

Soutenue le 20 décembre 2001 à l'Université devant un jury composé de :

| | | | |
|---|---|---|---|
| M | **Olivier** | **FAUGERAS** | Président |
| M | **Philippe** | **CINQUIN** | Rapporteurs |
| M | **Ron** | **KIMMEL** | |
| M | **Laurent** | **COHEN** | Directeur |
| Mme | **Françoise** | **DIBOS** | Examinateurs |
| M | **Sherif** | **MAKRAM-EBEID** | |
| M | **Wiro** | **NIESSEN** | |
| M | **Nicolas** | **ROUGON** | |

*A toi*

*( J. Dassin*, 1938-1980*)*

# Remerciements

Je tiens à remercier tout d'abord Messieurs Philippe Cinquin et Ron Kimmel pour avoir accepté le travail de rapporteurs de cette thèse. Monsieur Cinquin a été le premier à me faire entrevoir les possibles applications mathématiques au domaine médical, lors d'une présentation des activités de son laboratoire. En ce qui concerne Monsieur Kimmel

אני מודה מאוד את רון קמל על כך שעשה דרך ארוכה כדי להשתתף בועדת הבוחנים

Madame Françoise Dibos, qui m'a il y a bien longtemps enseigné les principes fondamentaux du traitement du signal, a accepté de participer à ce jury. Je tiens à l'en remercier, tout comme Monsieur Olivier Faugeras, pionnier de la vision par ordinateur, dont l'influence s'étend aussi maintenant au domaine médical.

Monsieur Nicolas Rougon, figure omniprésente du traitement d'image et des modèles déformables, me fait aussi l'honneur d'être membre du jury.

Ik zou in het bijzonder een dankwoord willen richten aan mijnheer Wiro Niessen, die helemaal uit het hoge noorden hierheen is gekomen. Het is voor mij een eer om een specialist inzake medische beeldverwerking in de jury te hebben.

Pour m'avoir guidé dans mes recherches tout au long de cette thèse, et pour m'avoir laissé aller dans les directions qui m'interessaient, pour m'avoir appris à communiquer sur mon travail et encouragé à publier, je tiens à remercier Monsieur Laurent Cohen. Merci de m'avoir fait confiance, et de m'avoir poussé en avant.

Monsieur Sherif Makram-Ebeid me fait l'honneur et l'amitié d'être lui aussi membre du jury. Ses compétences autant théoriques que techniques ont toujours été pour moi digne du plus grand respect. Son dynamisme et son aisance avec l'ensemble des théories de la vision (et autres) resteront pour moi un modèle du genre.

Je souhaite remercier toute l'équipe du groupe MedISys de Philips Recherche France, pour m'avoir accueilli en son sein depuis déjà plus de trois ans. La participation des gens de cette équipe à mon travail ne se limite pas à l'enseignement que j'ai pu tirer de leurs compétences et aux nombreuses contributions de chacun à mon ouvrage. Au dela de la bonne humeur et de l'ambiance sympathique, j'ai vraiment apprécié de faire partie de ce groupe: Cyrille Allouche, Odile Bonnefous, Jacques Breitenstein, Antoine Collet-Billon, Claude Cohen-Bacrie, Eric Denis, Maxim Fradkin, Raoul Florent, Olivier Gérard, Laurence Germond, Yves Hily, Françoise Iritz, Marie Jacob, Jean-Michel Lagrange, Pierre Lelong, Claire Lévrier, Claude Méquio, Lucille Nosjean, Jean Pergrale, Jean-Michel Rouet, Michel Siméon, et Nicolas Villain. J'en profite pour remercier Sandra Delcorso pour m'avoir soutentu pendant ces derniers mois de travail.

Pour ceux qui ont quitté le groupe entre temps, je souhaiterais distribuer quelques remerciements, notamment à Jean-Michel Létang, qui a été mon "parrain" au sein de l'équipe, et a guidé mes premiers pas dans le domaine de la recherche industrielle, tout en me faisant bénéficier de sa rigueur et sa créativité. Une mention très spéciale revient tout naturellement à Xavier Merlhiot et aussi Myriam Greff, anciens stagiaires du groupe: travailler avec vous a été plus qu'un plaisir, et je ne peux que reconnaitre l'influence primordiale de nos collaborations sur ma thèse, pour les "Ensembles de Niveaux" en ce qui concerne Xavier, et le "Live-Wire" pour Myriam. Vos compétences sont grandes et j'ai pris grand plaisir à travailler avec vous, et à inclure dans ma thèse le fruit de ces collaborations.

Au sein de la galaxie Philips, je tiens à remercier l'équipe MIMIT-Advanced Developement de Frans Gerritsen, et en particulier Roel Truyen et Bert Verdonck, pour leur accueil chaleureux, et leur intéret pour mon travail qui s'est trouvé concrétisé dans le développement de certains prototypes, et dans plusieurs publications communes.

Mes derniers remerciements vont tout naturellement à ceux qui m'ont soutenu pendant ces longues années d'études, c'est à dire à mes parents et toute ma famille. Merci aussi à tous mes amis qui ont su supporter les baisses de moral d'un chercheur souvent égaré voir perdu entre deux raisonnements. Vous êtes trop nombreux pour tous vous citer, mais mes remerciements s'adressent au clan de Boulogne-Billancourt, dans son sens le plus large. Je tiens à remercier Hélène Sellier pour m'avoir laissé mettre à contribution son grand talent pour le dessin et autorisé à mettre des représentations de ses oeuvres au début de chacune de mes parties. Et enfin merci à ceux et celles qui ont su me donner la force et la motivation nécessaires pour terminer, qui ont su, du fait de leur soutient, de leur présence ou de leur simple existence, donner une signification et un but à tout cela.

Thomas Deschamps
Paris, Décembre 2001

# Contents

# Abbreviations

| | |
|---|---|
| 1D | one-dimensional |
| 2D | two-dimensional |
| 3D | three-dimensional |
| 3DRA | 3D rotational angiography |
| AAA | abdominal aortic aneurysm |
| CAF | cost assignment function |
| CNR | contrast-to-noise ratio |
| CT | computed tomography |
| CTA | computed tomography angiography |
| CRC | colorectal cancer |
| DSA | digital subtraction angiography |
| DSR | digital subtraction radiography |
| fMRI | functional magnetic resonance imaging |
| Gd-DTPA | gadopentetate dimeglumine |
| GWDT | gray-weighted distance transform |
| MDL | minimum description length |
| MIP | maximum intensity projection |
| MPR | multi planar reformatting |
| MRA | magnetic resonance angiography |
| MRI | magnetic resonance imaging |
| pixel | picture element |
| SNR | signal-to-noise ratio |
| TTP | Tissue Transition Projection |
| US | Ultrasound (Imaging) |
| VOI | Volume of Interest |
| voxel | volume element |
| XRA | X-ray angiography |

# Introduction

## Context

Since several decades, the number of medical images produced in the world is constantly increasing. Three-dimensional medical images are becoming more and more used by the clinicians and the volume of datasets produced by the hospitals has created the need for an efficient structure for managing and processing those datasets. If 2D images are still useful and have numerous applications, since their acquisition cost is limited, cost for 3D images tend to reduce, and the frequency of their use is increasing.

3D images are produced as succession of slices that the clinician must mentally stack, in order to reconstruct and understand the 3D information they provide. This fact leads to misunderstanding of the 3D anatomical objects, and most of the manual processing tools lead to loss of information, since one dimension is not taken into account.

Automatic understanding of medical images is the underlying framework of this PhD study. We worked upon segmentation of anatomical structures, in other words extraction from the image of an anatomical object of interest, in order to visually inspect it and to quantify the extent of its possible pathologies. This segmentation process consist in isolating visible structures by recognizing their contours. The number of available methods to achieve this task in medical imaging is constantly growing. We can roughly classify those segmentation tools in two categories: a direct approach which consists in applying operators directly working on the image information; a modeling approach which uses an *a priori* information, trying to match a pre-defined modelization to the image in order to extract the targeted objects.

More specifically, tubular objects can be extracted and visualized, by discriminating them in the image, on the basis of their particular shape. The best way to study this kind of objects is to extract both their shapes and underlying structures, namely their skeletons, in order to guide a virtual inspection inside them, or to measure their diameters for example.

Using geometrical optic techniques, and the theory of wave light propagation in continuous medium, we are going to study more precisely fast and efficient algorithms for tubular shape extraction, and their geometric primitives, the curves that define their skeletons. Our methods are specifically dedicated to the particular shape of those objects.

And we will build tools in order to fly through the objects, and to numerically

characterize the extent of possible pathologies. Those tools will help clinicians in taking decisions concerning surgical issues and treatments.

## Contents

In the first part of the thesis, we propose a study of different problems related to path extraction in 2D and 3D medical images. The first chapter is a review of different techniques already used in the domain of the minimal paths, in particular techniques related to the active contours model. We focus on the formulation derived from the geodesic active contours and on a fast scheme for minimal path extraction, based on the formalism of the Level-Sets. The second chapter contains most of the work that has been done in the minimal paths domain, in particular it concerns the reduction of the user interaction and of the computing cost needed, in order to ease the use of the methods. The third chapter introduces two applications of the techniques developed in the previous one. The first application concerns virtual endoscopy, where the path to be extracted is a trajectory for a virtual camera which moves within the 3D datasets, and the second application is the development of an interactive segmentation tool, for real-time path extraction in 2D images, using the same minimal path principles.

Similarly, the second part of the thesis focus on surface extraction, using the same paradigm of the Level-Sets, as done in the first part. Chapter 4 reviews applications of the Level-Sets formalism in 3D medical imaging. Chapter 5 contains several algorithms which aim are to improve several drawbacks of the Level-Sets, including: introduction of interactivity in the segmentation process in order to make it robust, and several ways to decrease the huge computing cost generated by those methods. Chapter 6 is dedicated to applications of those improvements to two medical imaging problems: first one is visualization of cerebral aneurysms, in order to derive the optimal surgical treatment to avoid disorders, like cerebral hemoragges that can occur; second application is visualization of the colon polyps. The more accurate the early detection of those tumors, the more efficient the treatment.

Third part of the manuscript focuses on the study of more specific anatomical structures: tree structures, like vascular or bronchial trees. Our goal is to optimize visualization and quantification of the pathologies of those objects. In chapter 7, we detail tools which enable to see and measure pathologies on the basis of surfaces and curves extracted in the images. Chapter 8 links all the previous algorithms, to extract, in a single process, the surface and the skeleton of the object considered, using our curve and shape extraction tools elaborated in the two preceding parts of the thesis. In this chapter, tools are specifically adapted to the particular topology of the category of object studied. This surface information and the tree centered structures inside our tubular objects finds a natural justification in chapter 9, where its applied to vascular tree extraction in contrast-enhanced medical images. Those techniques are also applied to a more complex problem: bronchial tree extraction in multislice CT scanner datasets. Another application presented concerns reconstruction of vascular tree in 2D and 3D angiographic medical images, with methods based on perceptual grouping techniques.

## Contributions

In the first part of the thesis, we have enhanced significantly path extraction through the following major contributions:

- extension to 3D of the method proposed by *Cohen and Kimmel* [34];
- reduction of the computing cost of this method;
- reduction of the user interaction for initialization;
- design of a new method to extract centered paths;
- creation of a 3D trajectory extraction for virtual endoscopy, currently integrated in a commercial product, after succeeding clinical validation;
- development of a training tool for interactive and real-time path extraction on basis of the *Live-Wire*.

In the second part of the thesis, major contributions are

- development of a fast algorithm for pre-segmentation, on the basis of the minimal path techniques, and in particular the *Fast-Marching* algorithm;
- developments of techniques to improve interactivity of the *Level-Sets* which are able to handle topology changes but so include any interactivity property;
- design of a collaborative approach, based on the *Fast-Marching* and the *Level-Sets*.

Moreover, these contributions find their justification in the last chapter of this part: fast segmentation and accurate visualization of the pathologies are needed for segmentation of the cerebral aneurysms, and the characterization of the colon polyps.

In the last part, we present contributions specifically dedicated to the extraction and the quantification of tubular and tree structures:

- we adapt the minimal path formalism in order to provide a fast pre-segmentation of the tube-shaped objects;
- we find how to obtain from a segmentation the set of useful trajectories for inspection of those tubular objects;
- we explain how to derive the tree structure from the trajectories previously extracted;

And finally, we detail applications of those methods to three different problems in medical imaging:

- segmentation of vascular trees with pathologies like aneurysms and stenoses;
- creation of new segmentation methods for the complicated problem of bronchial tree extraction;
- reconstruction of vascular trees in 3D contrast-enhanced angiographic images, using perceptual grouping techniques derived from the minimal path formulation.

# I

# Path Extraction


SELLIER

# Chapter 1

# Minimal Paths in Image Processing

**Résumé** — Les chemins minimaux sont construits à partir de l'analogie avec la propagation de la lumière dans un milieu avec un certain indice de réfraction, suivant le principe de *Pierre de Fermat*. On explique tout d'abord dans la section 1.1 le lien entre ces chemins de lumière et la théorie de la réfraction, en montrant au passage l'intéret de l'application de cette théorie au traitement d'images.
Dans la section 1.2, partant de la formulation classique des contours actifs de *Kass, Witkin, et Terzopoulos* [82], on passe à la formulation de chemin minimal, telle qu'elle a été présentée par *Cohen et Kimmel* [34].
Mettant en parallèle la formulation discrète donnée par *Dijkstra* [43] des chemins minimaux dans des graphes avec le formalisme de *Cohen et Kimmel* [34], on explicite dans la section 1.3 différentes méthodes d'extraction de chemins, ainsi que la définition des différents termes intervenant dans ce modèle: la force liée à l'image, ou *Potentiel*.
Dans la section 1.4, on étudie le role du terme qui permet de controler la longueur du chemin, et son influence sur la courbure de ce dernier.

**Abstract** — Minimal paths are built upon analogy with the theory of wave-light propagation in a medium with a refractive index, according to the principles of *Pierre de Fermat*. We first explain in section 1.1 the link between this minimal light paths and the refraction principle, emphasizing the interest for the use of minimal paths in image processing.
Starting from the classical formulation of the active contours called *snakes* of *Kass, Witkin, and Terzopoulos* [82], we extend in section‘1.2 to the formulation of the minimal path, as presented by *Cohen and Kimmel* [34].
Comparing the discrete version of the minimal paths given by *Dijkstra* [43] with the continuous equivalent formalism of *Cohen and Kimmel* [34], we detail in section 1.3 several implementations of extraction techniques, and the settings of parameters involved in the model, such as the image force, named *Potential*.
In section 1.4 we study the influence of the offset term which controls the length of the minimal path (and its curvature).

## 1.1 Minimal Paths theory

### 1.1.1 The minimal path in geometrical optic

In order to understand the underlying law of refraction, behind the minimal path principle, let us imagine a straight seashore, separating the sea from the beach, as shown in figure 1.1. A lifeguard sitting at a point A in the beach sees a girl drowning

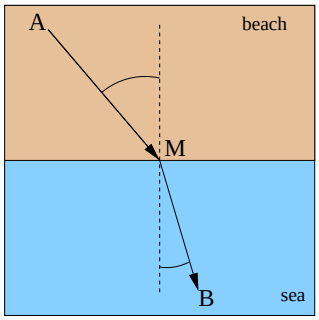



**Figure 1.1. Example of minimal path in a heterogeneous media**

at a point B in the sea. Assuming that the lifeguard can run on the beach three times faster than he can swim, the shortest time path is the broken line going straight from A to a point M on the shore, then straight from M to B. Considering the two angles of the straight lines to the normal to the shore, the ration of their sines is equal to the ratio of the corresponding speeds (*Descartes/Snell' law* of refraction). The principle of *Fermat* is that light waves of a given frequency travels the path between two points which takes the least time. The most obvious example of this is the passage of a light through a homogeneous medium, in which the speed of light does not change with position. In this case, shortest time is equivalent to the shortest distance between the points, which is a straight line, as shown in figure 1.2-left. When the medium is not homogeneous, as in figure 1.2-middle, there is a refraction angle at the interface between the two homogeneous regions. Figure 1.2-right can illustrate a well-known optical phenomenon, called *mirage*: The light source $S$ is visible from both points $R_1$, and $R_2$. But the light path between $S$ and $R_2$ is not a straight line, due to the difference of index of the two media. Therefore, $R_2$ "sees" S coming from location M, at the interface between the two media, while the image source is far from $M$. This phenomenon occurs when the variations in temperature are important enough to deviate the light path, resulting in "visions" of an oasis in the desert, for example. Hamilton defined optical path functions, which best known was defined by Burns as the *Eikonal equation* applied to the development of a mathematical theory of optical systems. The *Eikonal equation* is used to compute the minimal light paths for a refractive index, in the sense that the minimal path is the one which integral over the refractive index is minimal.

We are going to use this minimality property in order to extract curves in images, giving only the two extremities of the path, and using the equation developed by *Hamilton* for optical systems.

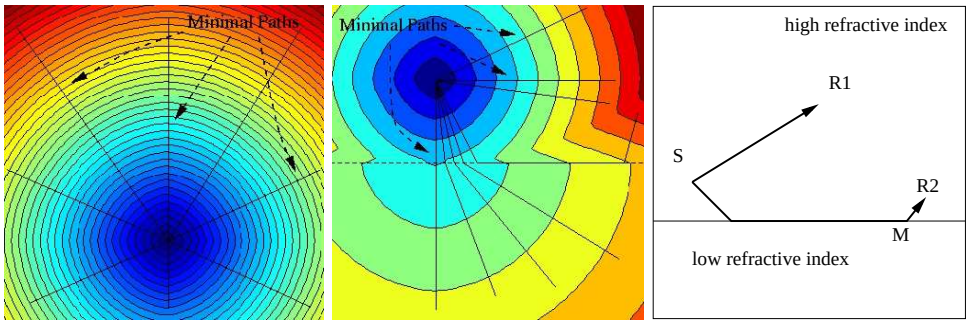



**Figure 1.2. Example of minimal paths in synthetic media:** left image represents the propagation of a light wave in a homogeneous medium, starting from a unique light source. The minimal paths between this light source and other position in the plane are straight lines. Middle image represents the propagation of a light wave in a medium where the refractive index is more important in the upper half part of the image than in the lower, starting from a light source in the upper part. Right image is a diagram which illustrates the *mirage* phenomenon on the basis of the propagation of the light in heterogeneous media, as shown in middle image.

### 1.1.2 Minimal path for curve extraction

We explain how this minimal path principle can be used for curve extraction in images. Given a refractive index $P$, called *potential* in the following, which takes lower values near the edges or features, our goal is to find a single contour that best fits the boundary of a given object or a line of interest. This contour, considered between two fixed extremities, will be the one which integral over the potential $P$ is minimal.

Looking for a path which lies in a desired region of interest, the refractive index should model the desired properties of the targeted curve. For example, in figure 1.3, we want to extract a path which stays inside the vessel. The dataset, a digital sub-

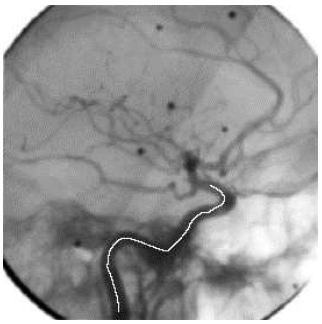


**Figure 1.3. Example of a minimal path in a media defined by a grey level image:** The minimal path (in white) superimposed on the date is the one that corresponds to the light wave propagation, using the grey-level information as a refractive index.

tracted angiography (DSA) shows vessels in lower grey levels on a bright background. Since we want to extract a path which stays inside the vessel, a possible refractive index to be used with the *Hamilton* equations for extracting minimal paths, could be the simple image grey level values.

This 'best fit' question leads to algorithms that seek for the minimal path, i.e. paths along which the integration over $P$ is minimal. Classical path extraction techniques are based on the *snakes* [82]. Snakes are a special case of deformable models as presented in [174]. Snakes start from a path close to the solution and converge to a local minimum of the energy. In this minimal path formulation, one interesting aspect is that the user input is limited to the end points, simplifying the initialization process. Unicity of this minimal path, for a given media avoids erroneous local minima. Motivated by the ideas put forward in [86,87] *Cohen and Kimmel* developed an efficient and consistent method to find the path of minimal cost between two points, using the surface of minimal action [87, 151, 178] and the fact that operating on a given potential (cost) function helps in finding the solution for our path of minimal action (also known as minimal geodesic, or path of minimal potential). In the following we show how the formulation of the minimal light path can be obtained through a modification of the classical formulation of the active contours, and we show the numerical implementation of the minimal path extraction.

## 1.2 The Cohen-Kimmel Method in 2D

Starting from the famous *Snakes* model, we show how can we derive a formulation of a minimal path extraction which leads to the expression of one of the optical equations of *Hamilton*, the *Eikonal*.

### 1.2.1 Global minimum for Active Contours

We present in this section the basic ideas of the method introduced by *Cohen and Kimmel* [34] to find the global minimum of the active contour energy using minimal paths. The energy to minimize is similar to classical deformable models (see [82]) where it combines smoothing terms and image features attraction term (Potential $P$):

$$E(C) = \int_{\Omega} \Big\{ w_1 \|C'(s)\|^2 + w_2 \|C''(s)\|^2 + P(C(s)) \Big\} ds \tag{1.1}$$

where $C(s)$ represents a curve drawn on a 2D image, $\Omega = [0, L]$ is its domain of definition, and $L$ is the length of the curve. Thus the curve is under the control of two kinds of forces:

- The internal forces (the first two terms) which impose the regularity on the curve. The choice of constants $w_1$ and $w_2$ determines the elasticity and rigidity of the curve.

- The image force (the potential term) pushes the curve to the significant lines which correspond to the desired attributes, for example

$$P(\mathcal{C}) = g(\|\nabla I(\mathcal{C})\|). \tag{1.2}$$

  Here, $I$ denotes the image and $g(\cdot)$ is a decreasing function. In the classical snakes [82], we have $g(x) = -x^2$. The curve is then attracted by the local minima of the potential, i.e. edges (see [61] for a more complete discussion of the relationship between minimizing the energy and locating contours). Other forces can be added to impose constraints defined by the user. As introduced in [29], previous local edge detection [20] might be taken into account as data for defining the potential.

The approach introduced in [34] modifies this energy in order to reduce the user initialization to setting the two end points of the contour $C$. They introduced a model which improves energy minimization because the problem is transformed in a way to find the global minimum. It avoids the solution being sticked in local minima. Let us explain each step of this method.

### 1.2.2 Problem formulation

Most of the classical deformable contours have no constraint on the parameterization $s$, thus allowing different parameterization of the contour $C$ to lead to different results. In [34], contrary to the classical snake model (but similarly to geodesic active contours), $s$ represents the arc-length parameter, which means that $\|C'(s)\| = 1$, leading to a new energy form. Considering a simplified energy model without any second derivative term leads to the expression $E(C) = \int \{w\|C'\|^2 + P(C)\}ds$. Assuming that $\|C'(s)\| = 1$ leads to the formulation

$$E(C) = \int_\Omega \{w + P(C(s))\}ds \tag{1.3}$$

We now have an expression in which the internal forces are included in the external potential. In [34], the authors have related this problem with the recently introduced paradigm of the level-set formulation. In particular, its Euler equation is equivalent to the geodesic active contours [23]. The regularization of this model is now achieved by the constant $w > 0$. This term integrates as $\int_\Omega wds = w \times \text{length}(C)$ and allows us to control the smoothness of the contour (see [34] for details). We remove the second order derivatives from the snake term, leading to a potential which only depends on the external forces, and on a regularization term $w$. It makes thus the problem easier to solve, and it is used in minimal paths [34], active contours using level sets [113] and geodesic active contours as well [23]. In [34] is also mentioned how the curvature of the minimal path is now controlled by the weight term $w$. This corresponds to a first order regularization term, and the paths show sometimes angles. A second order regularization term would give nicer paths, but it is difficult to include such a term in the approach.
Given a potential $P > 0$ that takes lower values near desired features, we are looking for paths along which the integral of $\tilde{P} = P + w$ is minimal. The surface of minimal

action $U$ is defined as the minimal energy integrated along a path between a starting point $p_0$ and any point $p$:

$$U(p) = \inf_{\mathcal{A}_{p_0,p}} E(C) = \inf_{\mathcal{A}_{p_0,p}} \left\{ \int_\Omega \tilde{P}(C(s))ds \right\} \tag{1.4}$$

where $\mathcal{A}_{p_0,p}$ is the set of all paths between $p_0$ and $p$. The minimal path between $p_0$ and any point $p_1$ in the image can be easily deduced from this action map. Assuming that potential $P$ is always positive, the action map will have only one local minimum which is the starting point $p_0$, and the minimal path can be found by a simple back-propagation on the energy map. Thus, contour initialization is reduced to the selection of the two extremities of the path.

It is possible to compute the surface $U$ in several ways. We are going to describe one of them that is consistent with the continuous case while implemented on a rectangular grid. It is, however, possible to implement a simple approximation like the shading from shape algorithm introduced in [178], or even graph search based algorithms ( [43, 151]), if consistency with the continuous case is not important.

## 1.3 Numerical Implementation

The numerical schemes we propose are consistent with the continuous propagation rule. The consistency condition guarantees that the solution converges to the true one as the grid is refined. This is known **not** to be the case in general graph search algorithms that suffer from *digitization bias* due to the *metrication error* when implemented on a grid [93, 119]. This gives a clear advantage to our method over minimal path estimation using graph search. Before we introduce the proposed method, let us review the graph search based methods that try to minimize the energy given in (1.3).

### 1.3.1 Graph Search Algorithms and Metrication Error

Based on the new energy definition (1.3), we are able to compute the final path without evolving an initial contour, by using the surface of minimal action. To find the surface of minimal action, graph search and dynamic programming techniques were often used, where the image pixels serve as vertices in a graph [26, 53, 122], considering the image as an oriented graph.

#### Oriented graph

A digital image is an array of pixels. With the optimal path approach, the pixel-array is considered to be a directed graph. A local cost is associated with each node of the graph, and a link cost is associated with every arc. The costs (both local and link) are determined by the energy to minimize, therefore also called cost-function. The problem of finding the optimal boundary segment between two image pixels is transformed into finding the optimal path between two nodes in the graph. A path between two points is said optimal if the sum of the cost function values at

each of its points (called cumulative cost) is lower than the cumulative costs of any other path between the same two points. Number of algorithms, based on dynamic programming, are available in the graph theory literature. Dijkstra's algorithm [43], which computes the optimal path between a single source point to any other point in the graph, is the basis of both the *Live-Wire* and the *Intelligent Scissors* 's graph search algorithms [49, 127].

#### Dijkstra's path search algorithm

Path extraction algorithms like *Live-Wire* [49] and *Intelligent Scissors* [127] tools use a search method based on Dijkstra's algorithm [43]. A description of Dijkstra's algorithm, applied to road detection, can be found in [53].

The algorithm (see table 1.1) is initialized by selecting a start point $s$ (also called seed point) in the graph. Giving this start point $s$, the cumulative cost of a pixel $p$ is the sum of the cost function values of the points of the optimal path from $s$ to $p$. The size of the active list is the total range of possible cumulative cost values, and its i-st item contains the list of pixels with their cumulative cost valuing i. The cumulative cost is initialized with value $\infty$ everywhere except at a start point $s$ with value zero. The active list is initialized by inserting the starting pixel into the first sub-list.

At each iteration of the algorithm, the pixel p with the lowest cumulative cost is removed from the active list and expanded: the cumulative cost of each of its non-expanded neighbors is updated, and the active list is reorganized. The updating of the cumulative cost of the neighbors runs as follows: the newly computed cumulative cost of a neighbor q is the sum of the cumulative cost of its father p and the link cost from p to q. If this newly computed cost is lower than the previous one the cumulative cost of q becomes the lately calculated cost, the active list is updated and a path map structure keeps a pointer from q back to p. Since at each iteration one pixel gets a final value, and a search for the minimal vertex to update is performed, the algorithm complexity is $O(N \log_2 N)$ where $N$ is the number of pixels in the image. In addition, a marker registers all the expanded pixels. The optimal path between the start point and another point in the image is obtained by following the pointers of the path map from the end point back to the seed point.

Our approach solves a continuous version of the problem. Sethian Fast Marching Method [161], described in section 1.3.2, has a similar complexity, yet it is consistent!

### 1.3.2 Fast Marching Resolution

In order to compute this map $U$, a front-propagation equation related to equation (1.4) is solved :

$$\frac{\partial C}{\partial t} = \frac{1}{\tilde{P}} \overrightarrow{n} \tag{1.5}$$

It evolves a front starting from an infinitesimal circle shape around $p_0$ until each point inside the image domain is assigned a value for $U$. The value of $U(p)$ is the time $t$ at which the front passes over the point $p$.

**Algorithm for optimal path search**

- Input:
  - cost function $\mathcal{P}$ ($\mathcal{P}(p,q)$ = cost of the arc $(p,q)$ in the graph);
  - starting seed pixel $s$;
- Output: the path map $\mathcal{M}$;
- Data structures:
  - cumulative cost array $CC$
  - Sorted active list $\mathcal{L}$
  - Boolean expanded pixels marker $\mathcal{E}$
- Begin:
  1. put $CC(s) = 0$ and $CC(n) = \infty\ \forall n$ in the set of graph vertices;
  2. put s in the list $\mathcal{L}$;
  3. while $\mathcal{L}$ is not empty do
     (a) consider $p$ the pixel in $\mathcal{L}$ with $CC(p)$ minimal;
     (b) consider $\mathcal{E}p$ = TRUE;
     (c) remove $p$ from $\mathcal{L}$;
     (d) For each neighbor $q$ of $p$ such that $\mathcal{E}p$ = FALSE, do
        i. $u = CC(p) + \mathcal{P}(p,q)$;
        ii. if $u < CC(q)$ then
           - $CC(q) = u$;
           - update position of $q$ in $\mathcal{L}$;
           - put a marker from $q$ to $p$ in $\mathcal{M}$.

**Table 1.1. Dijkstra Optimal Path Search Algorithm** as used in [49] and [127]

The *Fast Marching* technique, introduced in [2], [161], and detailed in [163], was used by [33], noticing that the map $U$ satisfies the Eikonal equation:

$$\|\nabla U\| = \tilde{P} \tag{1.6}$$

originally developed by *Hamilton and Burns* for geometrical optics (see section 1.1). Classic finite difference schemes for this equation tend to overshoot and are unstable. [163] has proposed a method which relies on a one-sided derivative that looks in the up-wind direction of the moving front, and thereby avoids the over-shooting associated with finite differences :

$$\begin{aligned} (\max\{u - U_{i-1,j}, u - U_{i+1,j}, 0\})^2 \quad &+ \\ (\max\{u - U_{i,j-1}, u - U_{i,j+1}, 0\})^2 \quad &= \quad \tilde{P}_{i,j}^2 \end{aligned} \tag{1.7}$$

giving the correct viscosity-solution $u$ for $U_{i,j}$. Authors of [150] also presented a direct numerical approach to solve (1.6) and gave a convergence proof to that minimization procedure in the viscosity solutions framework [35]. The principle of the *Fast Marching* is to introduce order in the selection of the grid points. This order is based on the

fact that information is propagating *outward*, because action can only grow due to the quadratic equation (1.7). Therefore the solution of equation (1.7) depends only on neighbors which have smaller values than $u$.
The algorithm is detailed in 3D in next section in table 2.1. The *Fast Marching* technique selects at each iteration the *Trial* point with minimum action value. This technique of considering at each step only the necessary set of grid points was originally introduced for the construction of minimum length paths in a graph between two given nodes in [43].
Thus it needs only one pass over the image. To perform efficiently these operations in minimum time, the *Trial* points are stored in a min-heap data-structure (see [1, 89, 160, 161, 163] for further details on the above algorithm, as well as a proof of correct construction). Since the complexity of the operation of changing the value of one element of the heap is bounded by a worst-case bottom-to-top proceeding of the tree in $O(\log_2 N)$, the total work is about $O(N \log_2 N)$ for the *Fast Marching* on a $N$ points grid.
Finding the shortest path between any point $p$ and the starting point $p_0$ is then simply done by back-propagation on the computed minimal action map. It consists in gradient descent on $U$ starting from $p$ until $p_0$ is reached, $p_0$ being its global minimum, since the geodesics are orthogonal to the wave fronts (see Bellman [11] for a nice proof on this orthogonality).

### 1.3.3 Back-propagation

In order to determine the minimal path between $p_0$ and $p_1$, we need only to calculate $U_0$ and then slide back on the surface $U_0$ from $(p_1, U_0(p_1))$ to $(p_0, 0)$. The surface of minimal action $U_0$ has a convex like behavior in the sense that starting from any point $(q, U_0(q))$ on the surface, and following the gradient descent direction, we will always converge to $p_0$. It means that $U_0$ has only one local minimum that is of course the global minimum and is reached at $p_0$ with value zero.

This is a consequence of the results in [11] that show that the light rays (geodesics, constant parameter curves) are orthogonal to the wave fronts (equal cost contours). The gradient of $U$ is therefore orthogonal to the wave fronts since these are its level sets. The back propagation procedure is a simple steepest gradient descent. It is possible to make a simple implementation on a rectangular grid: given a point $q = (i, j)$, the next point in the chain connecting $q$ to $p$ is selected to be the grid neighbor $(k, l)$ for which $U(k, l)$ is the minimal, and so forth.

### 1.3.4 Comparing Dijkstra and Eikonal

Since there is no difference in the overal complexity of both implementations, one may argue that using previously mentioned graph search algorithm like Dijkstra's [43, 155] might be sufficient. This algorithm is indeed efficient, but suffers from metrication errors. The graph based algorithms consider the image as an oriented graph in which each pixel is a node, and the 4 (or 8) connections to the neighboring pixels are the vertices of the graph. The different metrics used lead to very different results in a homogeneous media (see figure 1.4). Because in the $L_1$ metric considered for Dijkstra,

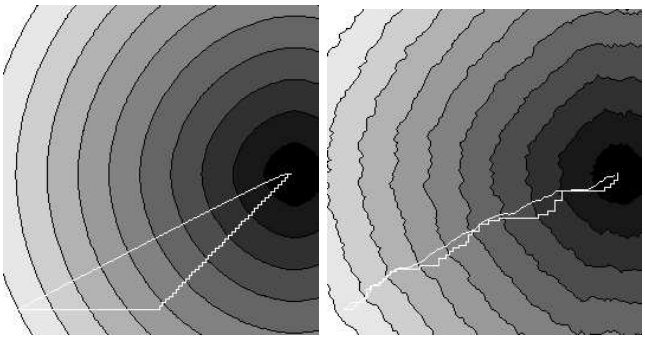


**Figure 1.4. Comparing Dijkstra and Eikonal optimal path extraction in a homogeneous media** Left image represents both paths with the iso-action levels. Right image is the same with an added Gaussian noise.

we are limited by the distance measure imposed by the graph, how fine the grid gets. With the 'right' Euclidean distance measure ($L^2$) we get the diagonal connection as the optimal path in this case. However, notice that the homogeneous media is a synthetic dataset, and that in a noisy image, both discrete and continuous formulation lead to similar results (see figure 1.4-right).

Of course the result of the graph-search could be improved by taking a larger neighborhood as structuring element [16, 176], but there will always be an error in some direction that will be invariant to the grid resolution, which is not the case in the discretization of the *Eikonal* equation.

This error is still too important for reasons of accuracy in medical applications for example. As an illustration, figure 1.5 shows the result of searching for the minimal path in a test image.

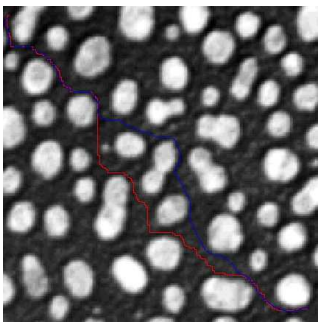


**Figure 1.5. Difference between Dijkstra and Eikonal on a real image:** The difference is obvious in this real image example. The potential used for computing the action is the grey level information. Using the same extremities, the red path corresponds to Dijkstra's, and the blue one to Eikonal.

## 1.4 The Regularity of the Path

In [34], it is proven that weight $w$ in equation (1.3) can influence curvature and be used as a smoothing term. An upper bound for the curvature magnitude $|\kappa|$ along the minimal path is found, $\mathcal{I}$ being the image domain:

$$|\kappa| \leq \frac{\sup_{\mathcal{I}} \|\nabla P\|}{w} \tag{1.8}$$

### 1.4.1 Influence on the gradient descent scheme

The exact minimal path is obtained with a gradient descent. But care must be paid on the choice of the gradient step to avoid oscillations.

If the weight $w$ is set to a small value $\epsilon$ the extracted path length is not limited at all, nor the curvature magnitude in equation (1.8). Therefore in zones where the action map is flattened, the slope being as small as $\epsilon$, the path can have a spaggethi-like trajectory. The minimal path being obtained by steepest gradient descent, directions are evaluated by interpolation based on nearest neighbors on the Cartesian grid. If the discrete gradient step $\Delta\mathbf{x}$ is too large, the approximation of this trajectory will produce oscillations between relative positions. Those oscillations can lead to a huge number of path points larger than forecasted allocations.

We have made a test on a region of the data shown in figure 1.6-left where the steepest gradient fails (with a number of path points limited). The cost map when

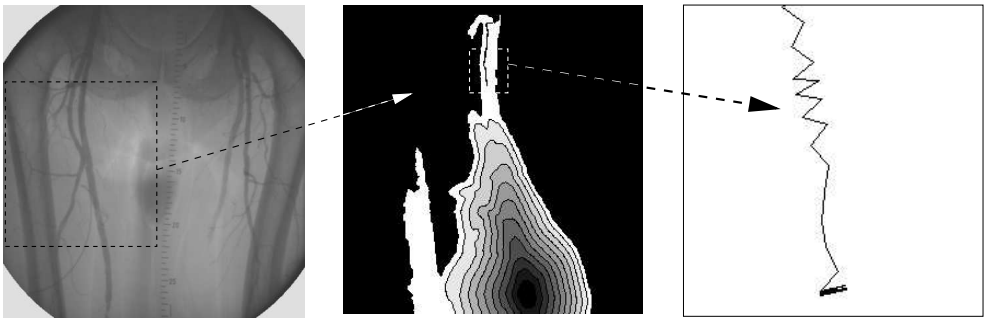


**Figure 1.6.** Failure of the steepest gradient descent on a bolus chase reconstruction data

tracking a vessel is displayed in figure 1.6-middle. Taking $w = .1$ leads to a curvature magnitude $\kappa \leq 10^3$. The steepest gradient scheme oscillates, for a given step size, and stops as shown in figure 1.6-right. Therefore, increasing $w$ maintains a lower upper-bound on the curvature magnitude and makes the steepest gradient descent scheme robust. Another method is to use more robust gradient descent techniques like Runge-Kutta where the step size of the gradient descent can be locally adapted.

### 1.4.2 Influence on the number of points visited

This section illustrates the influence of the weight $w$ of equation (1.3) on the necessary number of voxels visited for a path extraction. In figures 1.7 is shown the tracking of a vessel in a X-Ray image of the femoral vessels, using different weights $w_1 = 1$ and $w_2 = 20$. The smoothing done by increasing the weight can be observed in a

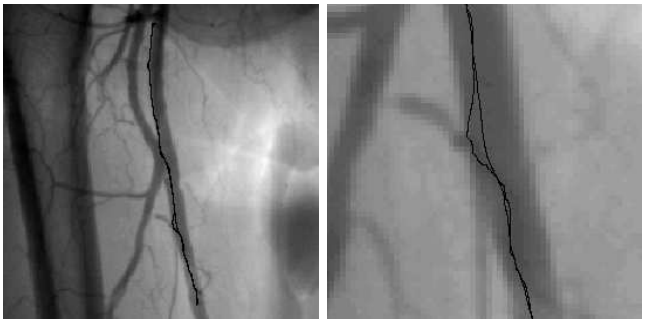
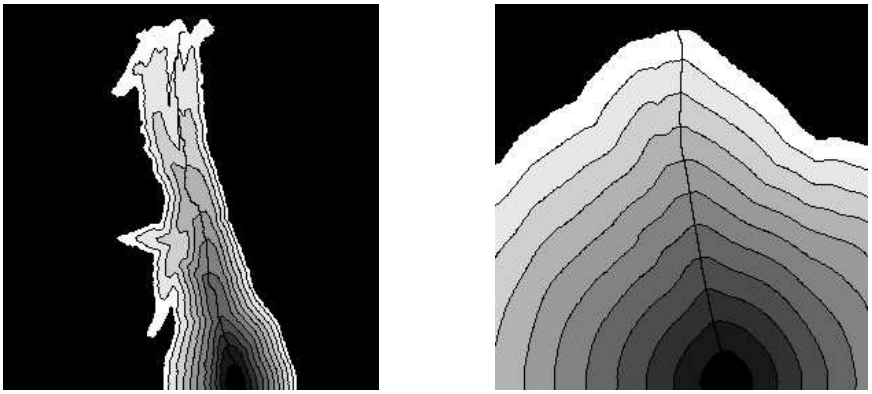


**Figure 1.7. Smoothing the minimal path with the weight $w$:** Left image shows the dataset of femoral vessels with two paths superimposed. Right image shows a zoom on the paths with $w_1 = 1$ and $w_2 = 20$.

zoom on the paths shown in figure 1.7-right. We can also observe the influence of increasing the weight in figure 1.8 where each path is displayed superimposed on its respective action map. For a small weight $w_1 = 1$, the path is not smoothed, as shown



$w_1 = 1$, and $|\kappa_1| \leq 15.4$ $w_2 = 20$, and $|\kappa_2| \leq 0.75$

**Figure 1.8.** Smoothing the minimal path with the weight $w$: the action maps

in figure 1.8-left. For a weight $w_2 = 20$, leading to the inequality $|\kappa_2| \leq 0.75$, the path is smooth. Differences appear also in the sets of points visited during propagations: it is smaller with weight $w_1 = 1$. It means that propagation is quicker for small weights. It propagates in every directions for a higher weight (see figure 1.8-right), because the tune of $w$ smoothes the image, as it reduces the upper-bound on curvature magnitude

in equation (1.8).

# Chapter 2

# Path extraction techniques based on the *Fast-Marching* algorithm

**Résumé** — Ce chapitre contient diverses améliorations de la méthode originale du chapitre 1, valables aussi bien en 2D qu'en 3D.
Nous commençons par présenter en section 2.1 une extension au 3D de la méthode classique. Nous détaillons en section 2.2 des méthodes pour accélerer l'extraction de chemins et la rendre plus facile, en réduisant les interactions nécessaires. Nous developpons une méthode pour extraire des chemins centrés dans des structures tubulaires en section 2.3. En section 2.4 nous calculons des trajectoires pour des objets en mouvement, en introduisant l'angle comme dimension supplémentaire au problème. Finalement, nous expliquons l'introduction d'un facteur de récursivité dans le *Fast-Marching* , afin d'extraire des chemins plus longs. Chaque technique est illustrée par un exemple sur une image réelle ou de synthèse.

**Abstract** — This chapter will detail various improvements and modification of the original method of chapter 1 that are useful for image analysis in 2D as well as in 3D.
In section 2.1, we extend the method detailed in section 1.3 for 2D images to 3D. In section 2.2 we give details about our techniques to make the path extraction scheme faster and easier, by reducing the user interaction. In section 2.3 we develop a new method to extract a path centered in a tubular structure. In section 2.4 we compute trajectories for moving objects, introducing a degree of freedom on their angulation. Finally, in section 2.5, we explain the introduction of a recursivity factor in the *Fast-Marching* algorithm, which enables to extract longer paths. Synthetic and real medical images are used to illustrate each contribution.

## 2.1 3D Extension

We are interested in this section in finding a minimal curve in a 3D image. One application that can motivate this problem is detailed in section 3.1. It can also have many other applications. Our approach is to extend the minimal path method of previous section to finding a path $C(s)$ in a 3D image minimizing the energy:

$$\int_{\Omega} \tilde{P}(C(s))ds \tag{2.1}$$

where $\Omega = [0, L]$, $L$ being the length of the curve. An important advantage of level-set methods is to naturally extend to 3D. We first extend the *Fast Marching* method to 3D to compute the minimal action $U$. We then introduce different improvements for finding the path of minimal action between two points in 3D. In the examples that illustrate the approach, we see various ways of defining the potential $P$.

Similarly to previous section, the minimal action $U$ is defined as

$$U(p) = \inf_{\mathcal{A}_{p_0,p}} \left\{ \int_{\Omega} \tilde{P}(C(s))ds \right\} \tag{2.2}$$

where $\mathcal{A}_{p_0,p}$ is now the set of all 3D paths between $p_0$ and $p$. Given a start point $p_0$, in order to compute $U$ we start from an initial infinitesimal front around $p_0$. The 2D scheme equation (1.7) is extended to 3D, leading to the scheme

$$\begin{array}{lcl} (\max\{u - U_{i-1,j,k}, u - U_{i+1,j,k}, 0\})^2 & + & \\ (\max\{u - U_{i,j-1,k}, u - U_{i,j+1,k}, 0\})^2 & + & \\ (\max\{u - U_{i,j,k-1}, u - U_{i,j,k+1}, 0\})^2 & = & \tilde{P}^2_{i,j,k} \end{array} \tag{2.3}$$

giving the correct viscosity-solution $u$ for $U_{i,j,k}$. The algorithm which gives the order of selection of the points in the image is detailed in table 2.1. It should be noted that a generalization of this algorithm was recently introduced in [92].

Considering the neighbors of grid point $(i, j, k)$ in 6-connexity, we study the solution of the equation (2.3). We note $\{A_1, A_2\}$, $\{B_1, B_2\}$ and $\{C_1, C_2\}$ the three couples of opposite neighbors such that we get the ordering $U_{A_1} \leq U_{A_2}$, $U_{B_1} \leq U_{B_2}$, $U_{C_1} \leq U_{C_2}$, and $U_{A_1} \leq U_{B_1} \leq U_{C_1}$. To solve the equation, three different cases are to be examined sequentially in table 2.2. We thus extend the *Fast Marching* method, introduced in by *Adalsteinsson and Sethian* [2], and used by *Cohen and Kimmel* [34] to our 3D problem.

## 2.2 Several minimal path extraction techniques

In this section, different minimal path extraction procedures are detailed. We present new back-propagation techniques for speeding up extraction, a one end-point path extraction method to reduce the need for interaction, and in next section, a centering path extraction method adapted to the problem of tubular structures in images. The methods presented in this section are valid in 2D as well as in 3D and this is an

**Algorithm for 3D Fast Marching**

- Definition:
  - *Alive* is the set of all grid points at which the action value has been reached and will not be changed;
  - *Trial* is the set of next grid points (6-connexity neighbors) to be examined and for which an estimate of $U$ has been computed using equation 2.3;
  - *Far* is the set of all other grid points, for which there is not yet an estimate for $U$;
- Initialization:
  - *Alive* set is confined to the starting point $p_0$, with $U(p_0) = 0$;
  - *Trial* is confined to the six neighbors $p$ of $p_0$ with initial value $U(p) = \tilde{P}(p)$;
  - *Far* is the set of all other grid points $p$ with $U(p) = \infty$;
- Loop:
  - Let $(i_{min}, j_{min}, k_{min})$ be the *Trial* point with the smallest action $U$;
  - Move it from the *Trial* to the *Alive* set (i.e. $U_{i_{min},j_{min},k_{min}}$ is frozen);
  - For each neighbor $(i, j, k)$ (6-connexity in 3D) of $(i_{min}, j_{min}, k_{min})$:
    - If $(i, j, k)$ is *Far*, add it to the *Trial* set and compute $U$ using table 2.2;
    - If $(i, j, k)$ is *Trial*, recompute the action $U_{i,j,k}$, and update it.

**Table 2.1.** *Fast Marching* algorithm

important contribution that can be useful for image analysis in general, for example in radar applications [8], in road detection [117], or in finding shortest paths on surfaces [86].

Examples in 2D are used to make the following ideas easier to understand. We also illustrate the ideas of this section on two synthetic examples of 3D front propagation in figures 2.1 and 2.2. Examples of minimal paths in 3D real images are presented in chapter 3.

The minimal action map $U$ computed according to the discretization scheme of equation (2.2) is similar to convex, in the sense that its only local minimum is the global minimum found at the front propagation start point $p_0$ where $U(p_0) = 0$. The gradient of $U$ is orthogonal to the propagating fronts since these are its level sets. Therefore, the minimal action path between any point $p$ and the start point $p_0$ is found by sliding back the map $U$ until it converges to $p_0$. It can be done with a simple steepest gradient descent, with a predefined descent step, on the minimal action map $U$, choosing

$$p_{n+1} = p_n - \text{step} \times \nabla U(p_n). \tag{2.6}$$

More precise gradient descent methods like Runge-Kutta midpoint algorithm or Heun's

**Algorithm for 3D Up-Wind Scheme**

1. Considering that we have $u \geq U_{C_1} \geq U_{B_1} \geq U_{A_1}$, the equation derived is
$$(u - U_{A_1})^2 + (u - U_{B_1})^2 + (u - U_{C_1})^2 = \tilde{P}^2 \quad (2.4)$$
Computing the discriminant $\Delta_1$ of equation (2.4) we have two possibilities
   - If $\Delta_1 \geq 0$, $u$ should be the largest solution of equation (2.4);
     - If the hypothesis $u > U_{C_1}$ is wrong, go to 2;
     - If this value is larger than $U_{C_1}$, go to 4;
   - If $\Delta_1 < 0$, at least one of the neighbors $A_1$, $B_1$ or $C_1$, has an action too large to influence the solution. It means that the hypothesis $u > U_{C_1}$ is false. Go to 2;
2. Considering that we have $u \geq U_{B_1} \geq U_{A_1}$ and $u < U_{C_1}$ , the new equation derived is
$$(u - U_{A_1})^2 + (u - U_{B_1})^2 = P^2 \quad (2.5)$$
Computing the discriminant $\Delta_2$ of equation (2.5) we have two possibilities
   - If $\Delta_2 \geq 0$, $u$ should be the largest solution of equation (2.5);
     - If the hypothesis $u > U_{B_1}$ is wrong, go to 3;
     - If this value is larger than $U_{B_1}$, go to 4;
   - If $\Delta_2 < 0$, $B_1$ has an action too large to influence the solution. It means that $u > U_{B_1}$ is false. Go to 3;
3. Considering that we have $u < U_{B_1}$ and $u \geq U_{A_1}$, we finally have $u = U_{A_1} + P$. Go to 4;
4. Return $u$.

**Table 2.2.** Solving locally the upwind scheme

method can be used for this path extraction. A simpler descent can be choosing $p_{n+1} = \min_{\{\text{neighbors of } p_n\}} U(p)$, but it gives an approximated path in the $L_1$ metric. Such a descent has no more the property of being consistent. As an example, see in figure 2.1 the computed minimal action map for a 3D Homogeneous medium defined by $P(i, j, k) = 1 \ \forall (i, j, k)$.

Figure 2.2 shows a front propagation an a synthetic binary example, based on a spiral. We extract a path that goes from the interior of the spiral, and finds its way out of it to the second end point outside the object.

### 2.2.1 Partial Front Propagation

An important issue concerning the back-propagation technique is to constrain the computations to the necessary set of pixels for one path construction. Finding several paths inside an image from the same seed point is an interesting task, but in case we have two fixed extremities as input for the path construction, it is not necessary to propagate the front on all the image domain, thus saving computing time. Figure 2.3 compares the sets of pixels visited using a classical front propagation, and a partial

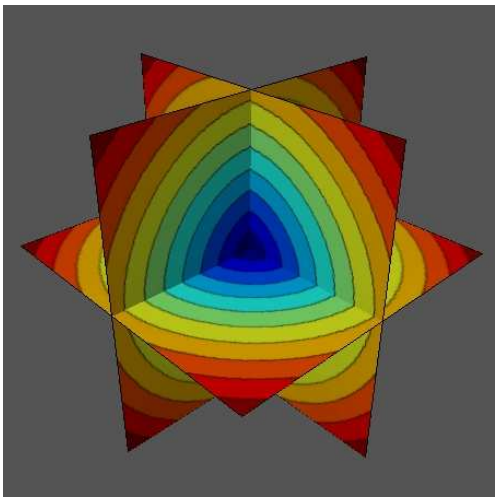


**Figure 2.1. 3D visualization of an action map:** these are the level sets of the action obtained by propagating a front in a homogeneous medium (constant Potential: $\forall(x,y,z) \in \mathbb{R}^3 \ P(x,y,z) = c > 0$) , represented with different colors. The domain is cubic, and the starting point for the wave equation is located at the center of the cube. The iso-action surfaces are concentric spheres with the starting point as center.

propagation on a Digital Subtracted angiography (DSA) image of the brain vessels. It highlight the fact that the set of points visited is smaller when propagation is only partial. We can see that there is no need to propagate further the points examined in figure 2.3-right, the path found being exactly the same as in figure 2.3-middle where front propagation is done on all the image domain. We used a potential $P(\mathbf{x}) = |\nabla G_\sigma * I(\mathbf{x})| + w$, where $I$ is the original image ($512^2$ pixels, displayed in figure 2.3-left), $G_\sigma$ a Gaussian filter of variance $\sigma = 2$, and $w = 1$ the weight of the model. In figure 2.3-right, the partial front propagation has visited less than half the image. This ratio depends mainly on the length of the path tracked.

### 2.2.2 Simultaneous Front Propagation

The idea is to propagate simultaneously a front from each end point $p_0$ and $p_1$. Let us consider the first grid point $p$ where those front collide. Since during propagation the action can only grow, propagation can be stopped at this step. Adjoining the two paths, respectively between $p_0$ and $p$, and $p_1$ and $p$, gives an approximation of the exact minimal action path between $p_0$ and $p_1$. Since $p$ is a grid point, the exact minimal path might not go through it, but in its neighborhood. Basically, there exists a real point $p^*$, which nearest neighbor on the Cartesian grid is $p$ which belongs to the minimal path. Therefore, the approximation done is sub-pixel and there is no need to propagate further. This *colliding fronts* method is described in table 2.3.

It has two interesting benefits for front propagation:

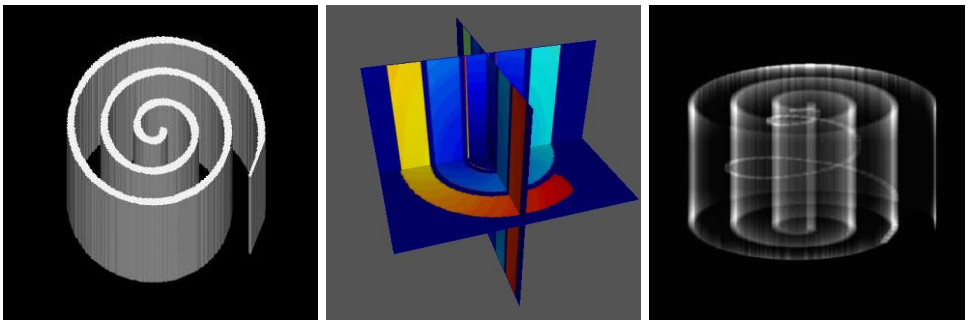


**Figure 2.2. Front propagation in a synthetic 3D example:** The left image is a volume rendering of the synthetic dataset, a spiral image, with a very high value ($P(x, y, z) = c_1 = 10^4$) in the spiral walls and a very low value in the background ($P(x, y, z) = c_2 = 1$). We extract the minimal path between a starting point located inside the spiral, and another one outside the spiral. The middle image represents the level sets of the action, mapped on three orthogonal planes, using the same color-map than in figure 2.1. The right image represents a transparent view of the object and the extracted path obtained.

**Algorithm**

- Compute the minimal action maps $U_0$ and $U_1$ to respectively $p_0$ and $p_1$ until they have an *Alive* point $p_2$ in common;
- Compute the minimal path between $p_0$ and $p_2$ by back-propagation on $U_0$ from $p_2$;
- Compute the minimal path between $p_1$ and $p$ by back-propagation on $U_1$ from $p_2$;
- Join the two paths found.

**Table 2.3.** Minimal Path as intersection of two action maps

- It allows a parallel implementation of the algorithm, dedicating a processor to each propagation;
- It decreases the number of pixels examined during a partial propagation. With a potential defined by $P = 1$, the action map is the Euclidean distance.
  - In 2D (figure 2.5), this number is divided by $\frac{(2R)^2}{2\times R^2} = 2$;
  - In 3D (figure 2.1), this number is divided by $\frac{(2R)^3}{2\times R^3} = 4$.

Figure 2.4 compares the sets of pixels visited using a partial front propagation, as explained in section 2.2.1, and a simultaneous propagation on a Digital Subtracted angiography (DSA) image of the brain vessels. It highlight the fact that the set of points visited is even smaller when propagation is done from both the extremities of the path.

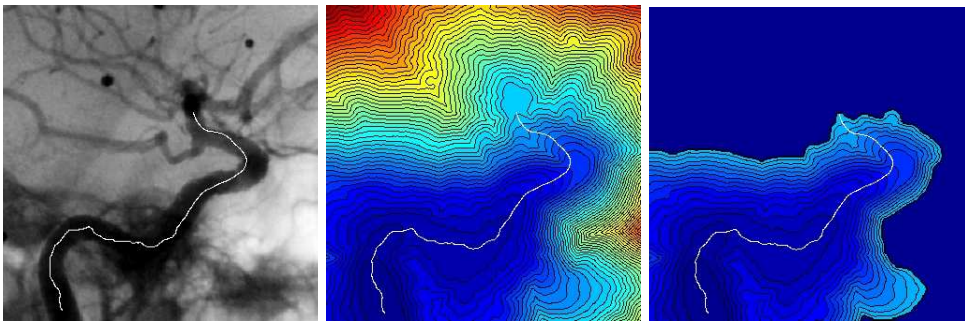


**Figure 2.3. Comparing classical and partial propagation:** The left image is the data set, used as potential to extract a path which stays inside a brain vessel of a DSA image; the extremities of the path are located manually. The center image is the action map obtained by propagating on the whole image domain. The right image shows the action map resulting from a partial computation. The two paths extracted are the same, due the minimality principle.

The potential used is $P(\mathbf{x}) = |I(\mathbf{x}) - C| + w$, where $I$ is the original image (256×256 pixels, displayed in figure 2.4-(a)), $C$ a constant term (mean value of the start and end points gray levels), and $w = 10$ the weight of the model. In figure 2.4-(b), the partial front propagation has visited up to 60% of the image. With a colliding fronts method, only 30% of the image is visited (see figure 2.4-(c)), and the difference between both paths found is sub-pixel (see figure 2.4-(d) where the paths superimposed on the data do not differ).

The diagram in figure 2.5 represents the theoretical difference of domains visited by the algorithm, for partial and simultaneous propagations.

### 2.2.3 Euclidean Path Length Computation

We have shown the ability of the front propagation techniques to compute the minimal path between two fixed points. In some cases, only one point should be necessary, or the needed user interaction for setting a second point is too tedious in a 3D image. Here we derive a method that builds a path given only one end point and a maximum path length.

As we explain below, we can compute simultaneously at each point the energy $U$ of the minimal path and its length. We choose as end point the first point for which the length of the minimal path has reached a given value. Since the front propagates faster along lower values of Potential, interesting paths are longer for a given value of $U$.

The technique is similar to that of section 2.2.1, but the new condition will be to stop propagation when the first path corresponding to a chosen Euclidean distance is extracted. Since the front propagates in a tubular structure, all the points for which the path length criterion is reached earlier in the process are located in the same area, far from the start point. Therefore the first point for which the length is reached is located in this area and is a valuable choice as endpoint.

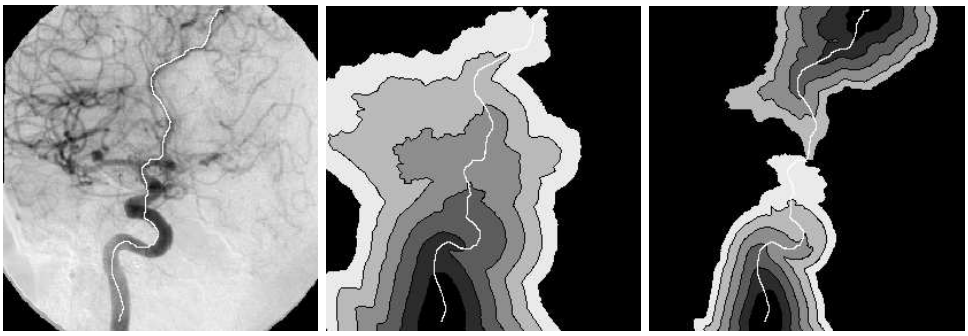


**Figure 2.4. Comparing partial and simultaneous propagation:** The left image is the data set, used as potential to extract a path in a vessel tree in a DSA image; the extremities of the path are located manually. The center image is the action map obtained by partial front propagating as explained in section 2.2.1. The right image shows the action map resulting from a simultaneous propagation from both extremities of the path. The second path extracted is a sub-pixel approximation of the first one, as detailed in the section.

Figure 2.6 represents the propagation of a front according to the potential shown in figure 2.6-left, with the *on-the-fly* computation of the approximate Euclidean length of the paths at each pixel crossed by the front. The propagation is done on the whole image domain, and one can observe that the resulting map, in figure 2.6-right is non-smoothed, and very difficult to analyze.

Figure 2.7 represents the same computation of the Euclidean path length than in figure 2.6-left, but limited to the necessary set of pixels visited in order to extract the minimal path super-imposed on the three images (the method is detailed in section 2.2.1). One can observe that the resulting map, in figure 2.6-right is non-smoothed, but we can clearly visualize the level-sets of the Euclidean path length computed at the same time.

## 2.3 Path centering in linear objects

The path is the set of locations that minimize the integral of the potential in equation (1.3). If the potential is constant in some areas, it will lead to the shortest Euclidean path. The same thing happens when the potential does not vary much inside a tubular shape. The minimal path extracted is often tangential to the edges, and would not be tuned for a problem which may require a centered path. Figure 2.8 describes the practical problem we are facing using the classical wave equation model of [34], in tube shaped structures where the potential is approximately homogeneous inside the object.

The general framework for obtaining a centered path is the following

- Segmentation : the first goal is to obtain the edges of the tubular region;
- Centered path : once we have this segmented region, we want to find a path

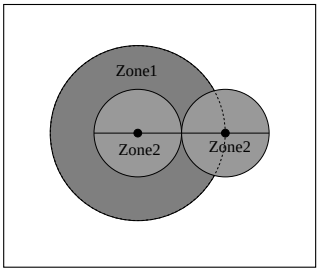



**Figure 2.5. Comparing the theoretical domains for extracting a minimal path between two points in a homogeneous medium:** The area labeled *zone 1* corresponds to the needed set of pixels visited with partial propagation. The area labeled *zone 2* corresponds to the needed set of pixels with simultaneous propagation.

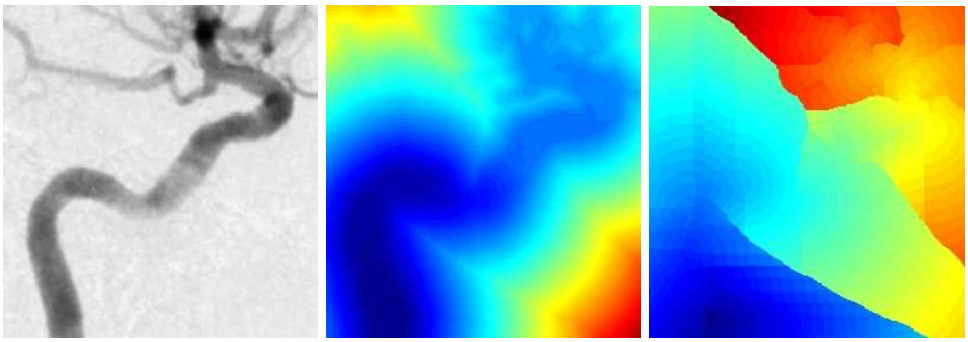


**Figure 2.6. Computing the approximate Euclidean path length while propagating on the whole image:** using the left image as potential, the front is propagated on the whole image domain. Middle image and right images represent respectively the action map starting from the bottom of the vessel, and the Euclidean path length computed at the same time.

that is as much centered as possible in it. In order to attract the minimal path to the center of the region, we use a distance map from the segmented edges.

In the following we are going to present our method, introduced in [42], detailing each step and making comparisons with other existing techniques.

### 2.3.1 Segmentation step

In order to find the tubular structure, several approaches can be used. We can use a balloon model [29] with a classical snake approach that inflates inside the object, starting with the given end point. Or we can segment the object using its correspondent level-sets implementation, as in [113] and like the bubbles in [172]. In fact, this kind of region growing method can also be implemented using the *Fast Marching* algorithm. This fast approximation has already been used for segmentation in [111].

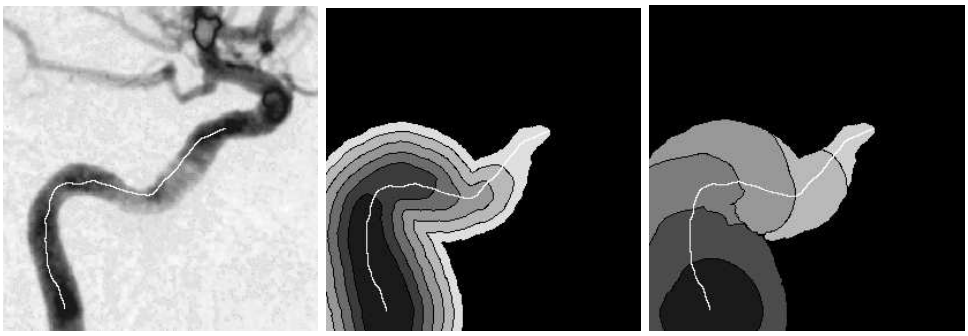


**Figure 2.7. Computing the approximate Euclidean path length while propagating partially on the image domain:** Left image is the potential. Middle image and right images represent respectively the action map starting from the bottom of the vessel, and the Euclidean path length computed at the same time.

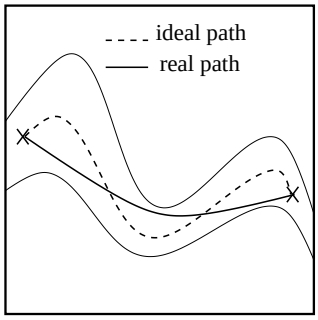



**Figure 2.8. Path extraction in a tube-like object:** the path obtained using the classical wave equation model is minimal according to the weighted metric, which means that it is the shortest in the tube considered; the ideal path would stay in the center of the object, as shown in the diagram.

This allows us to include the segmentation step in the same framework as our minimal path finding: having searched for the minimal action path between two given points, using a partial front propagation (see section 2.2.1), the algorithm provides different sets of points:

- the points whose action is set and labeled *Alive*;
- the points not examined during the propagation and labeled *Far*;
- the points at the interface between *Alive* and *Far* points, whose actions are not set, and labeled *Trial*.

This last category, the border of the visited points, is a contour in 2D and a surface in 3D which defines a connected set of pixels or voxels. If the potential is a lot higher along edges than it is inside the shape, the edges will act as an obstacle

to the propagation of the front. Therefore, the front propagation can be used as a segmentation procedure, recovering the object shapes. In this case the *Trial* points define a surface which can be described as a rough segmentation. Once the front has reached the endpoint, we use the front itself to define the edges.

### 2.3.2 Centering the path

Having obtained this interface of *Trial* points, we now want the information of distance to the edges. This information can be either used for a skeletonization, computing the medial-axis transform, or used as a new snake energy, that constrains the path in the center of the tubular shape.

In order to compute this distance, we can use a second front propagation procedure. The edges ares stored in the min-heap data-structure (see [163] for details), and this is a very fast re-initialization process to compute this distance. The potential and the initial action for this second front propagation are defined as follows:

$$\begin{aligned} P(i,j) &= 1 \quad && \forall (i,j) \text{ inside the shape} \\ P(i,j) &= \infty \quad && \forall (i,j) \text{ outside the shape} \\ U(i,j) &= 0 \quad && \forall (i,j) \in \{\textit{Trial}\} \text{ points of section 2.3.1} \\ U(i,j) &= \infty \quad && \text{elsewhere} \end{aligned}$$

Starting the front propagation from all the points stored in the min-heap data-structure, we compute the distance map, said $\mathcal{E}$, very quickly, visiting only the pixels inside the tubular object.

Our distance map $\mathcal{E}$ is used to create a second potential $P_1$. Choosing a value $d$ to be the minimum acceptable distance to the walls, we propose the following potential:

$$P_1(\mathbf{x}) = \max\left(d - \mathcal{E}(\mathbf{x}); 0\right)^{\gamma} \tag{2.7}$$

This distance $d$ is illustrated on figure 2.9. We use this potential (2.7) for a new front propagation approach: $P_1$ weights the points in order to propagate faster a new front in the center of the desired regions. This final propagation produces a path centered inside the tubular structure in a very fast process.

### 2.3.3 Description of the method

The complete method is described in figure 2.10.

1. Segmentation: the first step is to compute the weighted distance map given the start and end points. It is obtained by front propagation from the start to the end point. Notice also that the end point can be determined automatically by a length criterion as in section 2.2.3;

2. Segmentation: the second step is to consider the set of points which have same minimal action as the endpoint. For this, we store the front position (set of trial points) at the end of the first step.

3. Centering Potential : the third step is to compute the distance map $\mathcal{E}$ to the boundary front inside the tubular region. For this we propagate inward the front with a uniform potential $P = 1$. This gives the higher values towards the center of the object.

4. Centered path : the fourth step is to find the minimal path between start and end points relatively to the distance potential $P_1$ defined in (2.7) computed from the previous step. This is obtained by applying again the minimal path technique. The front is now pushed to propagate faster in the center of the object.

5. Centered path : the final step is to make back-propagation from the end point using the last minimal action map.

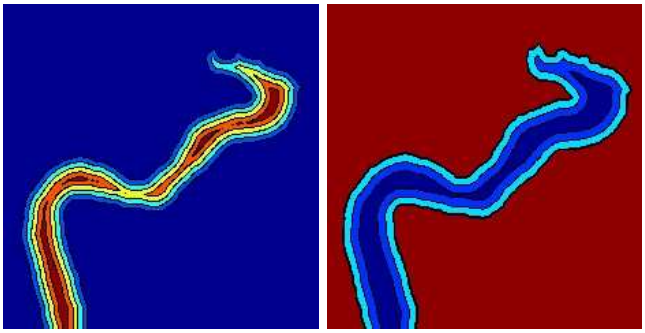


**Figure 2.9. Thresholding the distance map:** The left image shows the level sets of the distance to the object borders; The right image is the potential obtained by applying a threshold to the distance in order to propagate faster in a region of the tube, at a minimum distance to the borders, given as parameter.

Figure 2.10 explains the different steps of the path centering process.

An interesting improvement is that the value of the weight $w$ can be automatically set to a very low value:

- During the first propagation the regularity of the path is not important, and $w$ can be very small;
- During the second propagation, $P' = P + w = 1$
- During the final propagation the potential based on the distance to the object walls is synthetic and leads to smooth paths even if $w << 1$ .

Figure 2.11 compares the result of the classical path extraction, and the centering process detailed below, on a potential defined by a DSA image of the brain vessels. The two paths are represented super-imposed on the data in figure 2.11-left, in order to highlight the result of the modification of the penalty according to the distance to the object.

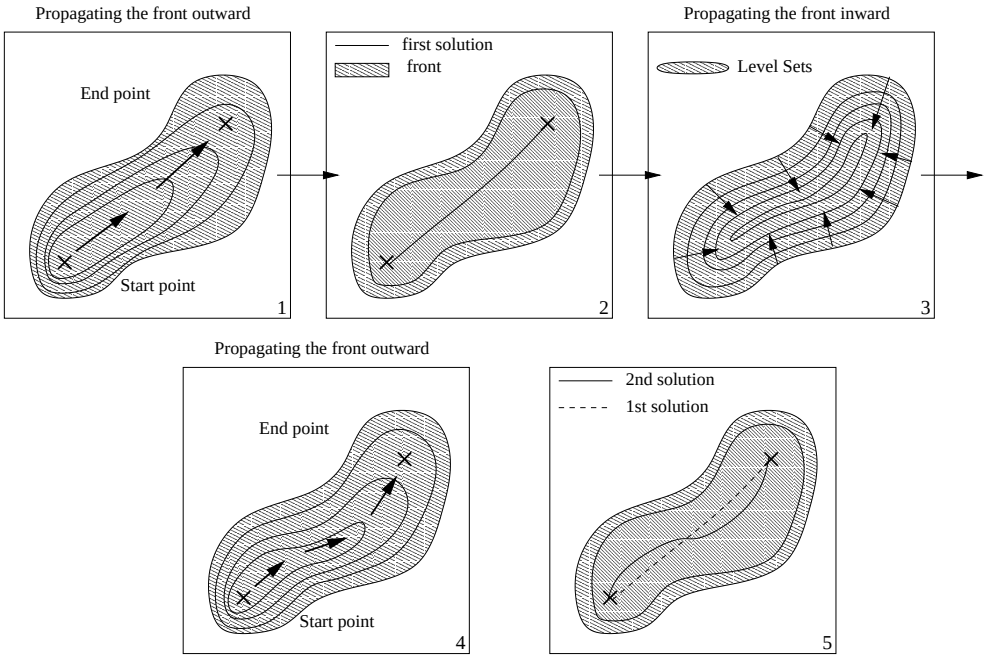



**Figure 2.10. The complete path centering process:** step 1 is propagation in the medium between the two extremities of the path; step 2 is to consider and store the envelope of the pixels visited during step 1; step3 is to propagate backward into the object, computing the distance to its borders; step 4 is to use the distance as a new penalty to propagate; step 5 is to backtrack the final centered path.

In figure 2.11-middle is shown the result obtained using a potential based on the image, where the shortest path is tangential to edges. But the front propagates only along the vessel direction, and is rapidly stopped transversally, allowing to compute the distance to the walls. Defining a new potential according to equation (2.7) based on this distance map, the second front propagates faster in the center of the vessel, at the distance $d$ chosen. Due to the shape of the iso-action lines of the centered minimal action shown in figure 2.11-right, the path avoids the edges and remains in the center of the vessel.

### 2.3.4 Comparison with other work

Another method to obtain a centered path would be to make a classical snake minimization on the centering potential $P_1$, starting from the path obtained previously, like it is done in the thesis [36]. But too much smoothing may lead to a wrong path. For example, in the case of thin tubular structures, smoothing the path may lead it outside the tubular structure. Also, the unpublished work presented in [36] details an algorithm which is applied to a tubular object which is already manually segmented by the user, whereas our method comprises both steps of segmentation and centering.

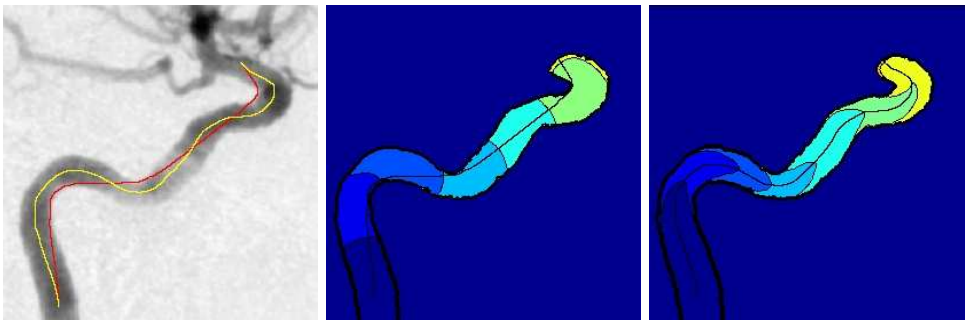


**Figure 2.11. Path centering test on a 2D DSA image:** Applying the process detailed in figure 2.10, the two paths extracted are super-imposed on the data on the left image; the middle image represents the level-sets of the action map with a classical use of the Eikonal equation; the right image shows the corresponding result with the centering strategy.

Another category of very similar centered line extraction technique is skeletonization, and particularly the definition of the medial axis function of [15] which treats all boundary pixels as point sources of a wave front. Considering that the *Fast Marching* computes the Euclidean distance to an arbitrary set of points using a potential $P = 1$, it can also be used for skeletonization.

However, the purpose of our application is to have a smooth line which always stays inside the tubular object and which is far from the edges.

If one wishes to achieve this task with a skeletonization, like in [192], he will need and rely on the results of post-processing techniques in order to obtain a unique and smooth path inside this segmented object. Smoothing and removing undesirable small parts of the skeleton can be done using techniques shown in [173]. The main advantage of our approach is that it gives only one smooth and centered path in a unique and fast process. Therefore, it cannot be replaced by a simple medial-axis transform.

In [136], the authors extract first the surface of the colon, then compute a minimal path on this surface and move this initial path to the center of the object by applying a thinning algorithm to the object segmented and projecting the path on the resulting surface. The algorithm developed by [90] can be applied to their methods since it computes the minimal path on a surface defined by a manifold. Although it seems to produce a smooth centered line, the thinning algorithm is computationally inefficient, compared to the speed of our algorithm that needs less than a minute on a classical inexpensive computer (300MHz CPU).

In the different techniques quoted, the main difference with our method lies in the fact that the object is manually segmented by the user. Our method comprises steps of segmentation and path extraction, and achieves them in a very fast way. More than a robust and fast method, we have developed a tool that is used for segmentation,

minimal path tracking, and even potential definition The main advantage of our approach is that it comprises all those steps and gives only one smooth and centered path in a unique and fast process.

## 2.4 Introducing the angle as a dimension

### 2.4.1 Principles

In [91, 92] the authors consider the problem of robot navigation with constraints and rotation, introducing a third degree of freedom in two-dimensional applications. Considering now an object with a given length and width, the problem is now to extract a trajectory between two positions that are in configuration space the position of the center of the object, plus an angle $\theta$ between 0 and $2\pi$ at the beginning and at the end of the trajectory.

The authors consider two cases, both on constant potentials:

- In the absence of obstacles, the *Fast-Marching* can be applied in a straightforward manner, by discretizing the configuration space into a 3D grid, namely griding both $\mathbb{R}^2$ and $[0; 2\pi]$ with periodic boundaries in $\theta$, and solving

$$\left[u_x^2 + u_y^2 + u_\theta^2\right]^{\frac{1}{2}} = 1 \tag{2.8}$$

- In the presence of obstacles, by altering the shapes of the obstacles for every discretized angle $\theta_i$, rather than maneuvering the robot. They use morphological operations, like dilatation to alter the obstacles shapes, with a structuring element corresponding to the robot at a given angle. A fast implementation of these morphological operations can be found in [64].

Therefore the above path planning problem solves the Eikonal equation

$$|\nabla T| = \frac{1}{V(x, y, \theta)} \tag{2.9}$$

where $F$ is binary: 1 in reachable regions and 0 inside the obstacles. As shown in [163], there is no reason to limit ourselves to binary speed values. We may use the same algorithm for continuously varying speed functions.

In [88], they consider the problem of obstacle avoidance navigating under a potential function which penalizes the free work space [100].

We worked upon the use of the Eikonal equation, including a dimension related to the angle of an object, in order to compute trajectories of oriented objects in domains with a weighted metric, without obstacles.

The problem has been schematized to the following:

1. We have used very simple objects, like rectangles and triangles, in two dimensional media;

2. For those objects, our strategy is to discretize them in a limited number of positions, which means that for a triangle, we only consider the value of the potential at its vertices.

3. In order to make the objects move according to the weighted metric, staying in the desired regions, at each position $(x, y, \theta)$, we take as potential for an object, the maximum of the potential over all positions considered (i.e. vertices for the triangle case).

We want to simulate the trajectory of an object, in a two dimensional medium, with constraints on the direction of the object: there is now a cost for changing its orientation. This result finds its application in the regularization of the point of view of the object in the media, simulating for example the direction of a virtual camera.

### 2.4.2 Algorithmic tricks

The algorithm used is very similar to that of section 2.1, which means we are working on a three dimensional problem. The following definitions are necessary:

1. The speed of the front is a function of the position, but not the orientation: $V(x, y, \theta) = V(x, y)\ \forall (x, y, \theta) \in \mathbb{R}^2 \times [0; 2\pi]$;
2. The action computed according to equation (2.9) is defined on $\mathbb{R}^2 \times [0; 2\pi]$, with periodic boundaries in $\theta$.
3. the griding of the interval $[0; 2\pi]$ used in the tests was to consider 10 discretized angle; it can be easily implemented using an array.

### 2.4.3 Results

Figure 2.12 represents samples of the trajectory of a triangle, using a DSA image as weighted metric for propagating. Obviously, the object is doing several U-turns along the trajectory and is not suitable for the applications we want to address.

We overcome the drawback of the very simple object used in figure 2.12-left by simply adding a branch to our triangle. This branch defines a new position for estimating the potential, and will constrain our object to look in the direction of the trajectory, in linear structures, like vessels. Results of this new strategy are shown in figure 2.12-right.

### 2.4.4 Perspectives

Linear objects with self-intersections: in a 2D X-ray image, a linear three dimensional structure projection can self-intersects, like a catheter in a heart image (for example see figure 2.13). The minimal path using only the 2D spatial configuration will not extract the loop which is created. Our method could overcome this drawback.

## 2.5 Introducing recursivity in the Eikonal equation

The *Fast-Marching* algorithm fails if the penalty is noisy, or if the objects to detect are long thin curves, like the guide-wire shown in figure 2.14. If the offset term $w$ and the penalty $\mathcal{P}$ in *Eikonal equation* are not tuned efficiently, a portion of the shortest

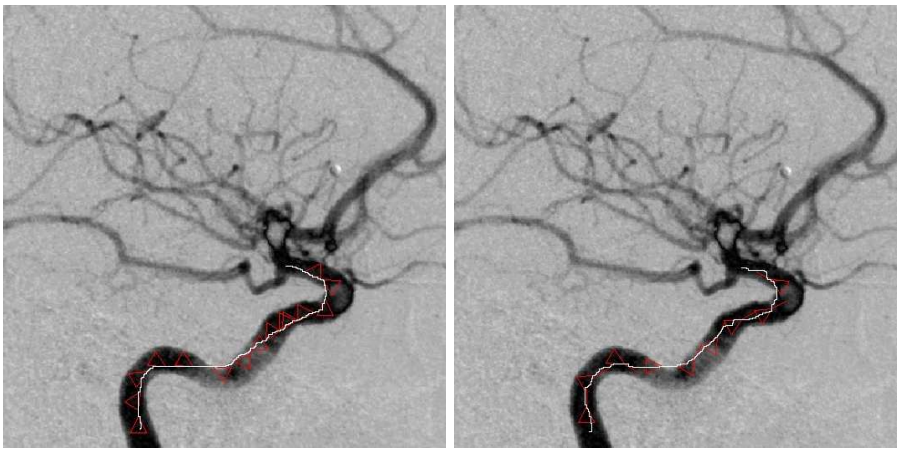


**Figure 2.12. Movement of a two different objects in the medium defined by the image on the background:** Left image: the object is a triangle; the path represents the trajectory of one of the vertices of the triangle. The constraint on the angle of the object is not sufficient, and the object is doing several U-turns during propagation; right image: the object is now a triangle with a branch connected at one of its vertices; the constraint on the angle of the object is now sufficient, and the object keeps looking in the direction of the trajectory.

path extracted can be a short cut to the starting point, leading to wrong results, like in figure 2.14-left. One way to overcome this drawback is to introduce a recursivity term in the computation in *Eikonal equation* : having $\alpha \in ]0;1]$, we now compute

$$\begin{aligned} (\max\{u - \alpha.\mathcal{U}_{i-1,j,k}, u - \alpha.\mathcal{U}_{i+1,j,k}, 0\})^2 &+ \\ (\max\{u - \alpha.\mathcal{U}_{i,j-1,k}, u - \alpha.\mathcal{U}_{i,j+1,k}, 0\})^2 &+ \\ (\max\{u - \alpha.\mathcal{U}_{i,j,k-1}, u - \alpha.\mathcal{U}_{i,j,k+1}, 0\})^2 &= \tilde{\mathcal{P}}^2_{i,j,k} \end{aligned} \tag{2.10}$$

giving the value $u$ for $\mathcal{U}_{i,j,k}$. This recursive term, usually set to .9, reduces the values of the action. This enables the front to propagate further in the direction of the thin curves, without propagating in all directions. In figure 2.14, the border of the set of visited points is drawn in red: on figure 2.14-left the algorithm has visited more than half the image domain, leading to a wrong path which goes straight down to the end point; on figure 2.14-right, the corresponding domain surrounds the guide-wire, and leads to a path that stay in the vicinity of the object.

Unfortunately, the equation( 2.10) of course no more gives the solution to any *Eikonal equation* computed on a penalty domain $\mathcal{P}$, and the new action map U computed is not convex at all. The minimal path that links the extremities is here defined by a $L_1$ descent on the map which stores the *Fast-Marching* iterations, and not with the gradient descent on the action map. The minimality principle is lost, whereas this algorithmic trick improves extraction. A patent has been filled on this subject (see [54]).

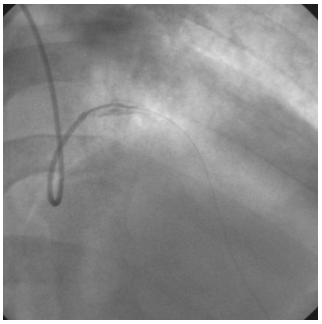


**Figure 2.13. Catheter in X-Ray image of the heart:** The catheter is the linear structures in dark that self-intersects.

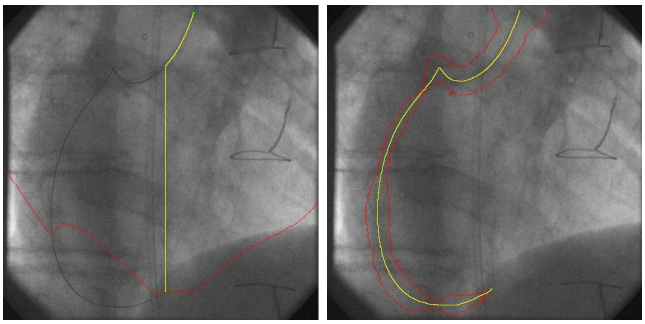


**Figure 2.14. Guide-wire extraction with recursive *Fast-Marching* :** The left image is the minimal path extraction with classical *Fast-Marching* ; right image is the same result with recursive *Fast-Marching* .

# Chapter 3

# Application to Virtual Endoscopy and to Several Problems in Medical Imaging

**Abstract** — Dans la première partie de ce chapitre on s'intéresse à la construction d'une méthode d'extraction automatique de chemins pour l'Endoscopie Virtuelle. C'est un procédé de navigation dans des images 3D, qui nécessite la définition d'une trajectoire précise pour l'observation en image de synthèse de l'intérieur du corps humain. Nous avons appliqué notre méthode d'extraction de chemins minimaux du chapitre 2 pour contruire ces trajectoires rapidement et de manière la plus automatique possible.
La seconde application concerne, au contraire, la construction de chemins de manière interactive, permettant à un utilisateur de dessiner rapidement les contours d'un objet, en ne précisant qu'un point de départ, sur le modèle des techniques appelées *Live-Wire* de *Falcao et Udupa* [49] ainsi que de *Mortensen et Barrett* [127]. Sur la base de notre méthode, nous avons développé un outil similaire, incluant une possibilité d'adapter les paramètres du modèle, au cours de la segmentation, en l'entrainant à reconnaitre les contours qui nous intéressent.

**Abstract** — First section concerns the creation of a fully automatic path tracker for *Virtual Endoscopy* . *Virtual Endoscopy* visualizes the inner surfaces of structures present in volumetric data in 3D images. As navigation through the inner structures quickly becomes a complicated procedure, especially when these structures are strongly curved, often a trajectory through the structure is used. We applied our path extraction technique developed in chapter 2 in order to build an accurate and fast tool to automatically builds those trajectories.
For the second application, we have focused on the need in many applications, as in medical imaging, for interactive segmentation, offering the possibility to a non-expert to draw quickly the boundary of an object, following *Live-Wire* technique of *Falcao and Udupa* [49], and *Mortensen and Barrett* [127]. *Live-Wire* methods restrict the intervention of the user to the setting of a start point in an image. Using our path extraction algorithms, we have developed the same tool, including a training method that adapt the different parameters of the model *On-The-Fly* .

## 3.1 *Virtual Endoscopy*

Visualization of volumetric medical image data plays a crucial part for diagnosis and therapy planning. The better the anatomy and the pathology are understood, the more efficiently one can operate with low risk. Various post-processing techniques have been available, to enable the radiologist to recognize a pathological condition, in the shortest amount of time: three 2D orthogonal views (see figure 3.1), maximum intensity projection (MIP, and its variants, see figure 3.2), surface and volume rendering.

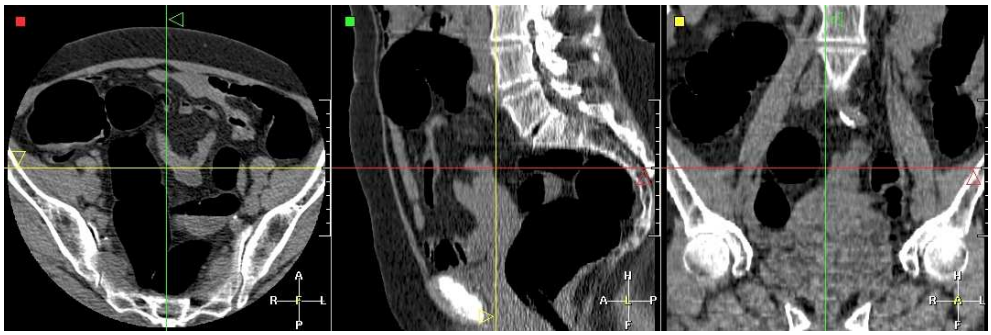


**Figure 3.1. 3D CT scanner of the colon:** Display of the dataset of $512 \times 512 \times 121$ voxels using three orthogonal views; air was injected in the colon (dark regions of the image) before acquisition in order to inflate the object and increase accuracy of the reconstruction of the colon.

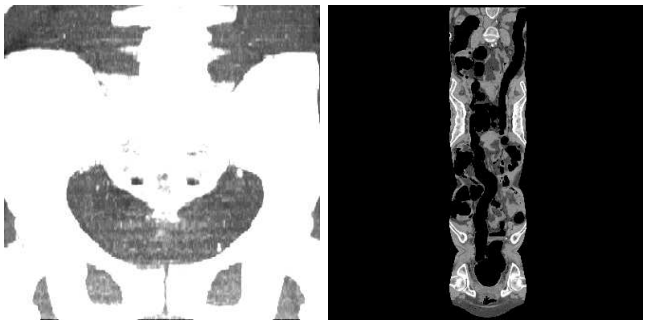


**Figure 3.2. Different rendering techniques of a 3D CT scanner of the colon:** left image is a Maximum intensity projection **MIP** and right image is a curved reformat image of the dataset shown in figure 3.1.

The maximum intensity projection requires to chose a direction of projection, therefore the volume is projected on a 2D plane, keeping only the brightest image points (in figure 3.2-left, the brightest intensities are given by the bones).

The curved reformat image in figure 3.2-right is a particular case of a multi-planar reformat image **MPR** : A **MPR** is a cross-sectional image of the 3D volume, along

an arbitrary plane, and a curvilinear reformat is a section along a surface generated by a bi-dimensional curve $f : \mathbb{R} \rightarrow \mathbb{R}, \mathrm{y} = \mathrm{f(x)}, \forall \mathrm{z} \in \mathbb{R}$. For a 3D trajectory, there are 3 different ways to represent the **MPR** , by fixing either $x$, $y$ or $z$. The path is placed in the target of investigation (the colon in figure 3.1), and the organ appears in the reconstruction in its full length. Advantage is to display the whole complex shape of the colon on a single 2D image, but the anatomical objects are distorted, and dimensions are not reliable on the final image.

*Virtual Endoscopy* allows by means of surface/volume rendering techniques to visually inspect regions of the body that are dangerous and/or impossible to reach physically with a camera (e.g behind an airway stenosis or obstruction, or too small). An extensive definition of *Virtual Endoscopy* can be found in [80, 146]. In chapter 7, we detail visualization techniques based on volume and surface renderings.

*Virtual Endoscopy* techniques can be divided into two groups of methods that can collaborate:

- techniques which deal with simulation of a real endoscope motion; In this case, *Virtual Endoscopy* is very interactive, simulating the motion of a camera inside the body, based on an extracted anatomical object that is modeled using rigid body dynamics; good examples of this simulation are presented in [77, 131].

- techniques which focus on the observation of the interior of anatomical objects by extracting trajectories inside them, see *Yeorong et al.* [192] for an example.

We have focused on the second kind of techniques. However, the minimal path techniques can also be useful for the first kind of methods: Kimmel and Sethian have applied the Fast-Marching algorithm for a robotic application in [163], for the motion of an object with a certain shape and orientation in an image with obstacles. We have also investigate this field in section 2.4, modeling the movement of an object, discretizing *Eikonal equation* in a space that describes the object position and orientation. But adding a dimension to the problem could lead to huge computing costs for an interactive 3D application.

Figure 3.3 is the usual framework that builds a *Virtual Endoscopy* facility, and is usually composed of two parts:

- A Path construction part, which provides the successive locations of the fly-through in the tubular structure of interest (see figure 3.3-left);

- Three dimensional interior viewing along the endoscopic path. Those views are adjoined creating an animation which simulates a virtual fly-through through them (see figure 3.3-right).

### 3.1.1 Medical relevance

In recent years, computerized post-processing techniques of image data from cross-sectional imaging modalities has received increasing recognition in the field of medicine. Technical developments of acquisition systems such as CT and MRI have improved along with continuously increasing spatial resolution. But if each axial image were

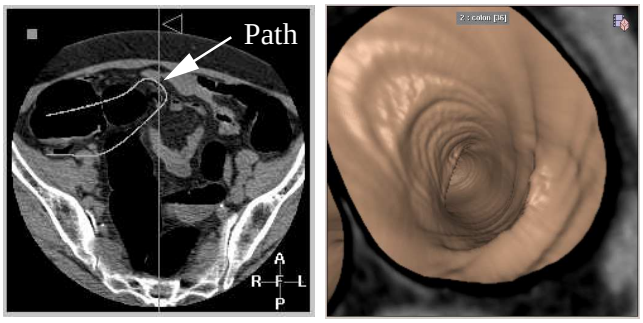



**Figure 3.3. *Virtual Endoscopy* framework:** Left image shows an example of trajectory manually drawn using the orthoviewer facility; right image is a volume rendering of the dataset, using a ramp function manually parameterized, at a position along this path.

to be viewed before one could proceed to the next image, the viewer would require an enormous amount of time to shift through all slices. At the same time, growing computer performance has created the opportunity to display anatomical structures in a comprehensible manner. *Virtual Endoscopy* is one of the most recent innovations in the spectrum of post-processing techniques. The predominant motive is to simulate conventional endoscopy, as in figure 3.4-left, by means of a non-invasive and safe technique, presenting the image data included in the original slices, in a movie-mode, in such a fashion that the radiologist is able to differentiate between that which is healthy and that which is pathological. Figure 3.4 shows how a clinician is able to detect pathologies using the volume rendering tool associated to the centered path extraction. But the true diagnostic performance of *Virtual Endoscopy* and the ability

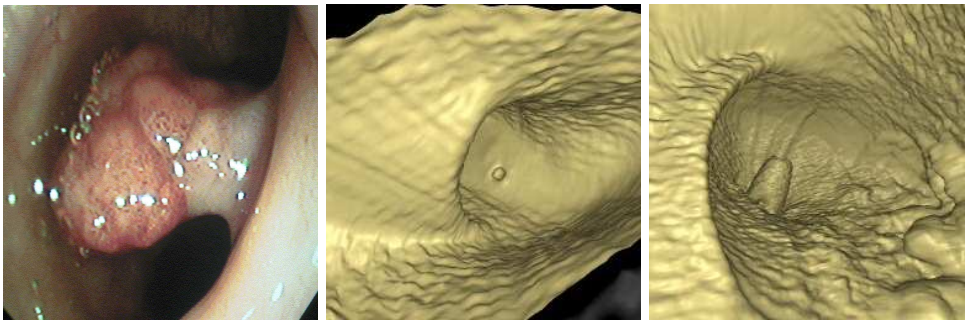


**Figure 3.4. Pathologies detected with the volume rendering:** Left image is a conventional endoscopy image of the colon; the middle image displays a small cavity which is a diverticulum in the colon surface; on the lower-right part of the right image, the bumps are polyps.

to reproduce acceptable results outside of research protocol has not been established yet. Clinical validation is still an on-going process, and researchers focus on the ap-

plication and validation of this technique for particular organs and pathologies. For colorectal cancer (CRC) (see figure 3.4-right), studies (see [146]) show that net effect of the use of *Virtual Endoscopy* could reduce cancer-related deaths in the future.

### 3.1.2 Problem with the path construction

A major drawback in general remains when the path construction is left to the user who manually has to "guide" the virtual endoscope/camera. The required interactivity can be very tedious for complex structures such as the colon for example. Actually, on most clinical platforms the user must define all path points manually, using for example three 2D orthogonal views, as shown in figure 3.1, leading to problems as the following:

- Since the anatomical objects have often complex shapes, they tend to pass in and out of the three orthogonal planes. Consequently the right location is accomplished by successively entering the projection of the desired point in each of the three planes;
- The path is approximated between the user defined points by lines or Bezier splines. If the number of points is not sufficient, it can easily cross an anatomical wall.

Path construction in 3D images is thus a very critical task and precise anatomical knowledge of the structure is needed to set a suitable trajectory, with the minimum required interactivity.

Numerous techniques try to automate this path construction process. Most of them use a skeletonization technique, like in [192], in order to extract a centerline in the dataset. But extracting the skeletons of an anatomical shape requires first to segment it. And the skeleton often consists in lots of discontinuous trajectories, and post-processing, as done by *Tek and Kimia* [173] is necessary to isolate and smooth the final path. The front propagation techniques studied in this application in contrast to other methods does not require any pre- or post-processing as explained in section 2.3.

It is sometimes necessary to smooth the path extracted by the front propagation. The point of view in the volume rendering of the tubular structure is very important, because it constrains the result of the examination. Thus, during the virtual fly through, the point of view of the camera must change smoothly. Traditionally, the position of the virtual camera frame at a particular path point is orthogonal to the path. If the path is not smooth, the point of view of the virtual camera will change in an abrupt manner. There are two ways to achieve this regularization:

- by modifying the view angle of the virtual camera, being no more orthogonal to the path, but looking in the direction of a path point which is located far from its current position, or using a running average of the local direction of the camera;
- by increasing the weight $w$ in equation (1.3) since it has a smoothing effect on the minimal path (see appendix for details). We preferred to use this technique in the following examples, since it is efficient and very simple to add.

### 3.1.3 Proposed solution for colonoscopy

The method detailed here is illustrated by results performed on the volumetric CT scan shown in figure 3.1.

#### Building a potential for virtual colonoscopy

The target is to build a potential $P$ with the 3D data set allowing paths to stay inside the anatomical shapes where end points are located. We thus define the potential by a general model $\tilde{P}(x) = |I(x) - I_{mean}|^{\alpha} + w$.

First, the potential must be lower inside the colon in order to propagate the front faster, and to avoid problems with crossing the edges of the anatomical object. In a colon CT scan, an average position $I_{mean}$ of the colon grey level in the histogram can be defined (see figure 3.5) as a peak in the histogram where $I_{mean} = 200$. Secondly,

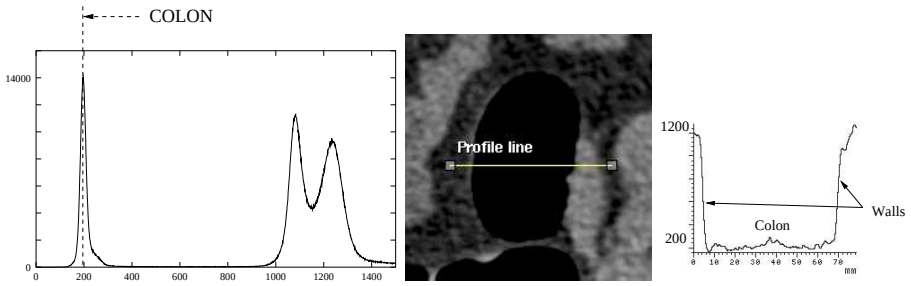



**Figure 3.5. Global and Local study of the image intensities:** Left image is the histogram of the whole 3D dataset where the colon can be clearly distinguished; middle image is a slice of the colon where a profile line was drawn in order to observe the variation of intensities that are displayed in right image.

if the path to be extracted is very long, the situation can lead to pathological cases, and the front can go through potential *walls*. This is frequent for large objects that have complex shapes and very thin edges, as colon. Then, edges should be enhanced to enable long trajectories, with a non-linear function. We thus take $\alpha = 2$ in order to enhance the dynamic of the image with a quadratic function.

However, this potential does not produce paths relevant for *Virtual Endoscopy* . Indeed, paths should remain not only in the anatomical object of interest but as far as possible from its edges. In order to achieve this target, we use the centering potential method as detailed in section 2.3. We first need to obtain a *shape* information. In fact, a CT scan of the colon contains already a shape information sufficient to constrain a front propagation. In figure 3.5-middle is shown a slice of a colon volumetric data set, and figure 3.5-right shows the corresponding grey level profile. Air fills the colon and is represented in our CT image by a grey level around 200 (see figure 3.5-right), while edges are defined by a grey intensity around 1200. Then, using the potential $\tilde{P}(x) = |I(x) - I_{mean}|^{\alpha} + w$, the front obtained through *Fast Marching* is stopped by the anatomical shapes. Figure 3.6 shows the propagation of the wave equation, according to the penalty defined previously. It also illustrates the fact that the *Fast-Marching* can act also as a segmentation tool.

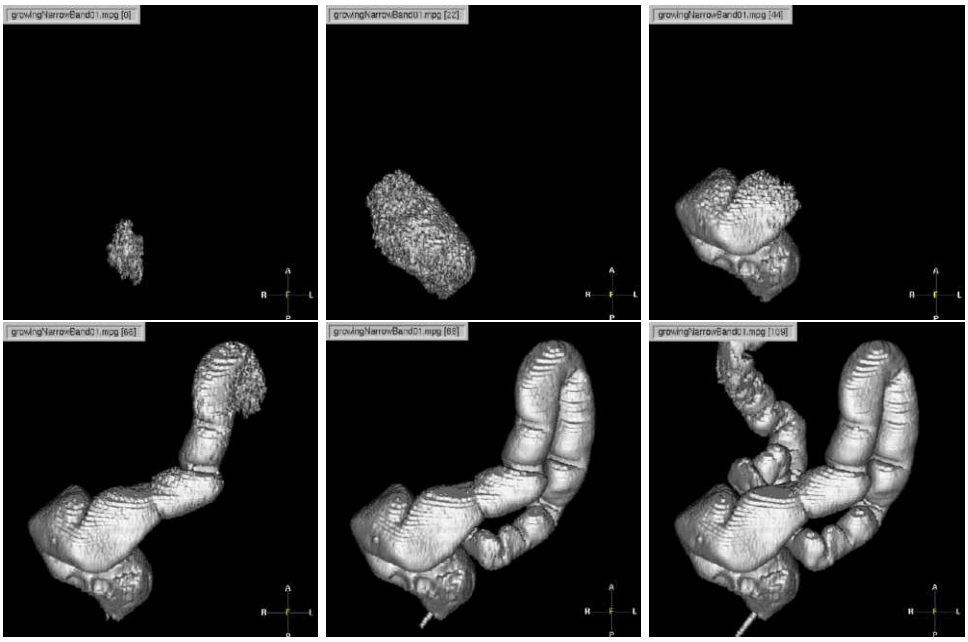


**Figure 3.6. Wave propagation inside the colon dataset:** these are volume renderings of the *Alive* points at different consecutive iterations during propagation.

### Path centering technique

The edges are obtained via a first propagation: in figure 3.6 we can see the evolution of the *narrow band* during propagation. It gives a rough segmentation of the colon and provides a good information and a fast re-initialization technique to compute the distance to the edges. Using this distance map as a potential (from equation (2.7)) that indicates the distance to the walls, we can correct the initial path as shown in figure 3.7-left: the new path remains more in the middle of the colon. And the value of the parameter $d$ can be derived from anatomical characteristics. If we know approximately the section of the colon along the path we can easily choose a value to stay in the center of the tubular structure.

The two different figures 3.7-middle and 3.7-right display the view of the interior of the colon from both paths shown in figure 3.7-left. With the initial potential, the path is near the wall, and we see the u-turn, whereas with the new path, the view is centered into the colon, giving a more correct view of the inside of the colon. The new centered path is smooth because this final propagation is done on a synthetic potential (the distance to the walls) where noise has been removed.

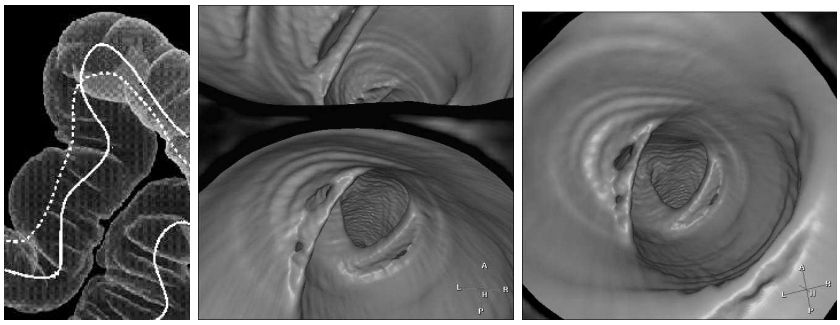


**Figure 3.7. Comparing normal and centered paths:** The left image is a Tissue Transition Projection (TTP) of the colon surface; middle image is the endoscopic view obtained with the classical trajectory (represented with a doted curve) in the U-turn shown in the left image; right image shows the endoscopic view resulting from the centered path (represented with a plain curve on left image) at the same position in the colon.

### 3.1.4 Results on objects filled with air

#### Results on Colonoscopy

Virtual colonoscopy is one of the most exciting developments in gastrointestinal radiology. It is a non-invasive, well-tolerated, and safe technique for evaluating colorectal cancer (CRC). **CRC** represents the third most frequently diagnosed cancer worldwide. For the United States, it is estimated that 129,000 new cases were diagnosed in 1999 [168]. Preliminary results indicate that the accuracy of virtual colonoscopy exceeds that of conventional endoscopy. Our automatic path extraction provides in a very fast process a path that remain inside the structure and avoid collisions with the wall, following in a smooth manner the centerline of the colon. Once this path is created, navigating is simply a matter of positioning the view along the path. Figure 3.8 shows the difference between the paths extracted with the classical method, and the centering method.

Notice that in figure 3.8-middle, the path has been smoothed by the centering method. This is mainly due to the fact that the noise inside the colon has been *eaten* by the first propagation. The small variations in intensity allows the front to propagate in all the colon, and considering the distance to the borders of the object *cleans* the anatomical object, by considering an artificial penalty $P(x, y, z) = 1\ \forall (x, y, z) \in \mathbb{N}^3 \cap \text{colon}$. Figure 3.8-right illustrates the complexity of the path extracted: intersecting the same slice several time, the user who wants to extract it manually must have strong skills in anatomy, and time. Figure 3.9 shows the corresponding endoscopic viewings generated with those two paths.

#### Results on the Trachea

The virtual inspection of the trachea is the same problem as in the colon. It is even simpler, due to the topology of the organ observed. Figure 3.10-left displays a 3D

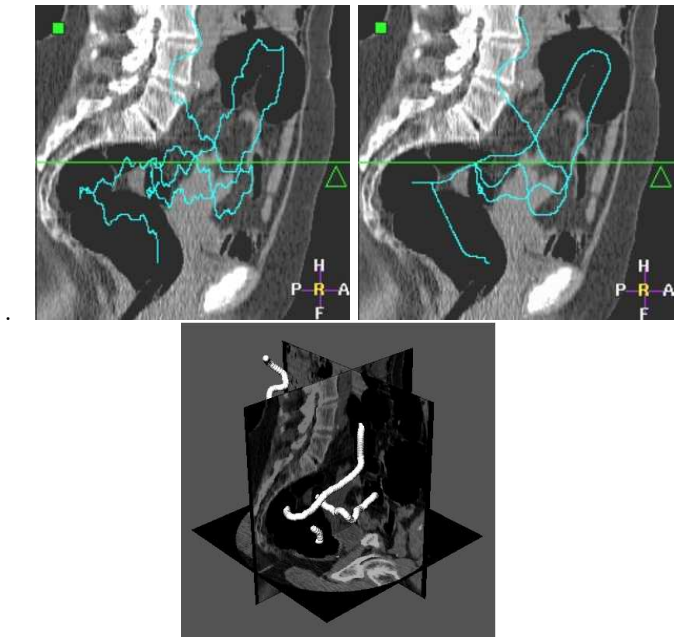



**Figure 3.8. Path extraction in the 3D CT scanner of the colon:** Left image is the superposition of a path extracted from only one point manually located inside the colon over a slice of the dataset; middle image displays the centered path in the same configuration; right image is the representation in 3D of the trajectory displayed in middle image intersecting three planes, where the dataset has been mapped.

CT dataset of a trachea with three orthogonal views. Air fills the object and gives a shape information all along from throat to lungs. Therefore, the anatomical object having a very simple shape, the path construction with one or two fixed points is easier than in the colon case. One example path tracks the trachea, using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 200|^2 + 1$). This path has been used to display the **MPR** view of figure 3.10-right. An endoscopic view along the same path is displayed in figure 3.11. The inspection of the trachea is often part of the inspection of the whole tracheobronchial tree. In part III, we will study more precisely the tracheobronchial tree. The examination of the complete tree needs new algorithms to extract the complete tree structures, and to segment the borders of the bronchi. This implies use of more complicated algorithms than *Fast-Marching* and will be further developed.

### 3.1.5 Results on Arteries and vessels

Is endoscopy a useful tool for artery and vessel examination?

The volume-based rendering tool use an opacity threshold which can create severe artifacts on the endoscopic viewings. And the quality of the images is a direct result of

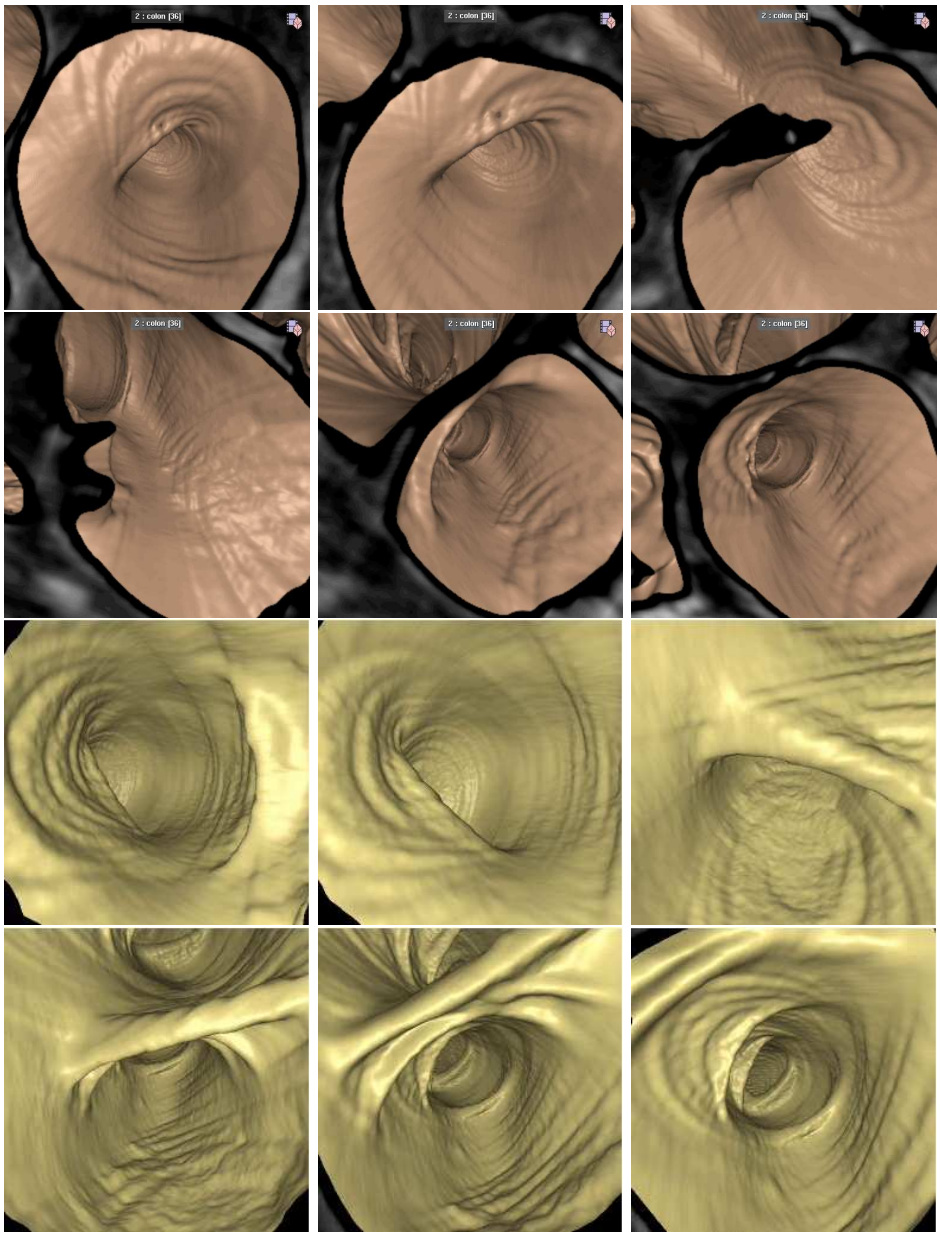


**Figure 3.9. *Virtual Endoscopy* in the colon, minimal and centered paths:** On the first two rows the trajectory being the shortest, the intersection plan of the camera shows adjacent structures and shrinks the validity of the examination; on the two last rows, the trajectory being centered, the point of view of the virtual camera shows the entire tube, and the validity of the examination is increased.

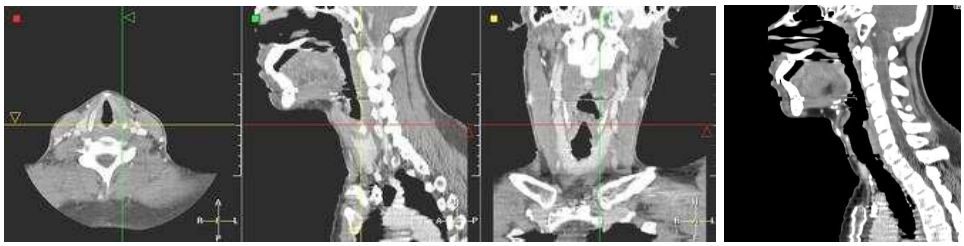


**Figure 3.10. 3D CT scanner of the trachea:** Left image is a dataset of 320 × 320 × 234 voxels; the interior of the trachea is very easy to characterize from the whole image since it is always filled with air; right image is a **MPR** image along the trajectory extracted.

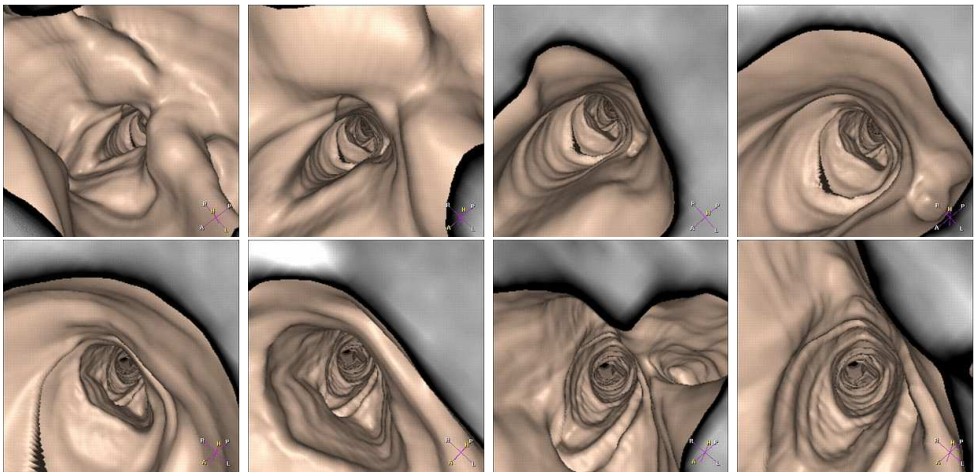


**Figure 3.11. *Virtual Endoscopy* in the Trachea:** starting from the mouth, the virtual camera goes straight to the bifurcations between the lungs, following the path extracted (see **MPR** view in figure 3.10-right).

the homogeneous opacification of the blood. The blood is made visible using contrast products, and this image information depends now on the section of the object. It can be also severely impaired in areas of turbulent flows.

Review of the axial slices alone is somehow sufficient for the diagnosis assessment of several pathologies. But, still automatic path extraction provides important visualization assessments on the datasets. It can be used with multi planar reformatting **MPR** images, which enable to see the tubular objects with optimal slice orientation, depicting spatial relationships between the object and its pathologies. But *Virtual Endoscopy* offers an intraluminal view of the arteries and veins and allows inspection of pathologies, like stenosis. It enhances visualization, by clearly showing the spatial relationship between the pathology and the object, comparing those pathologies before and after treatment, as done for stent placement in abdominal aortic aneurysm **AAA** . It can be also input in systems for the virtual planning of endoluminal aortic

stents (see [186]).

This gives justification for the following study on the automatic path extraction in veins and arteries. But the grey-level information is more complex in the case the signal is obtained through injection of a dye product in the object of interest. The boundaries of the object are more difficult to extract, therefore the path centering technique developed in section 2.3 and previously applied to the colon case, is no longer valid for the following. However, automatic path extraction has remained valid, despite the variability of the new penalty information. Centering techniques for those kind of objects involve the use of more complex algorithms, like achieved segmentation algorithms (see *McInerney and Terzopoulos* [115] for a review of medical image segmentation with deformable models), and have been later developed *a posteriori* in part III of the thesis.

### 3D MR image of the Aorta

A test was made on an aorta MR dataset. Figure 3.12-left displays this 3D MR dataset of the abdominal aorta using three orthogonal views. Contrast product was injected

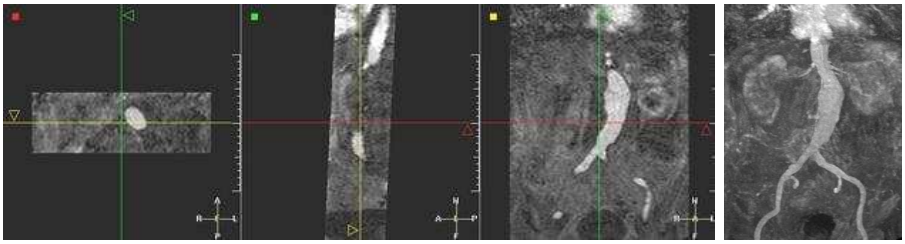


**Figure 3.12. 3D MR image of the aorta:** Dataset of $256 \times 256 \times 60$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the right image is a threshold based volume rendering of the aorta itself, which highlights that the anatomical object has a visible pathology: an Abdominal Aortic Aneurysm (AAA).

before acquisition. On the **MIP** view, in figure 3.12-right, the important variation of the section of the object, upon the bifurcation of the iliac arteries, clearly indicates an an Abdominal Aortic Aneurysm **AAA** . The dye fills the aorta, and makes it visible, among other soft tissues, while bones are not rendered. The propagation measure is based on a nonlinear function of the intensity of the contrast solution that fills the aorta. This data set is difficult since the intensity of the contrast product will vary along the aorta (the contrast bolus dilutes during the acquisition time). Due to this non-uniformity, paths can cross other anatomical structures with similar intensities if the mean value inside the aorta is not set correctly by the user. Our example path tracks one iliac artery (see figure 3.13), using the potential $\tilde{P}(x) = |I(x) - 1000|^2 + 10$ in the MR scan. The dataset contains noise, and we must use an important weight to smooth the extracted paths. We have displayed a sample of the endoscopic views of the aorta along the path in figure 3.14. During the virtual fly through, the observer clearly notice the important variation of the object cross-

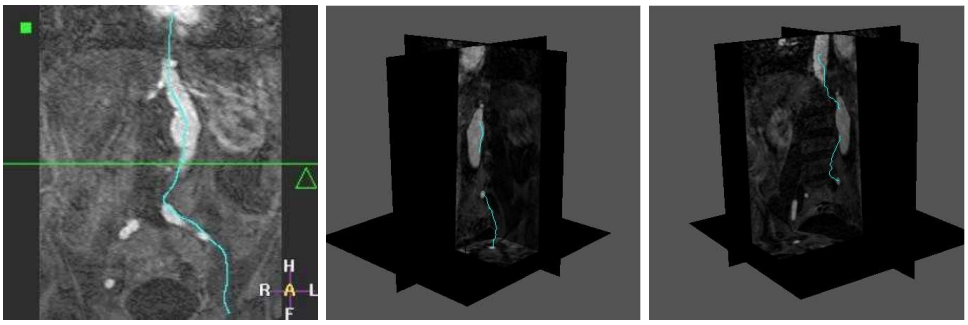



**Figure 3.13. Path extraction in the 3D MR image of the aorta:** Left image is the superposition of one path extracted between two points manually located inside the aorta over a slice of the dataset; middle image is a curved reformat sagittal image along one of the paths which tracks the right iliac artery; right image is a volume rendering view based on an opacity function parameterized manually along the same trajectory.

section. The movie in figure 3.14 depicts this pathology, even if the threshold-based volume rendering produces artifacts, when using the image grey level information obtained from a contrast enhanced MR image.

### 3D CT Scanner of the Aorta

Figure 3.15 displays a 3D CT dataset of the abdominal aorta using three orthogonal views. The contrast product injected before acquisition fills the aorta, and make it visible among other objects. The problem is similar to path extraction in a MR image, as done previously. Main difference here lies in the high resolution of the CT scanner dataset of figure 3.15, towards the MRA dataset of figure 3.12. This high resolution increases significantly the computing time of the method, but even if the dynamic of the images is very different, it does not lead to major differences in the parameterization of the path extraction process. Figure 3.16-left displays several paths extracted with the same seed point. Figure 3.16-middle is a curved reformat view along one of the trajectories extracted. This kind of view enables to validate trajectories by verifying that the section drawn always intersect the structure of interest.

### 3D MR image of the brain vessels

Tests were performed on brain vessels in a MRA scan. Three orthogonal slices of this dataset are shown in figure 3.17 together with a path extracted. The problem is different, because there is only signal from the dye in the cerebral blood vessels. All other structures have been removed. The main difficulty here lies in the variations of the dye intensity. The example path tracks the superior sagittal venous canal (the vein on the top of the head, as shown in figure 3.17-right), using a nonlinear function of the image grey levels ($\tilde{P}(x) = |I(x) - 100|^2 + 1$). Two views of the extracted path in 3D are displayed in figure 3.18 together with 3 orthogonal slices of the dataset. A

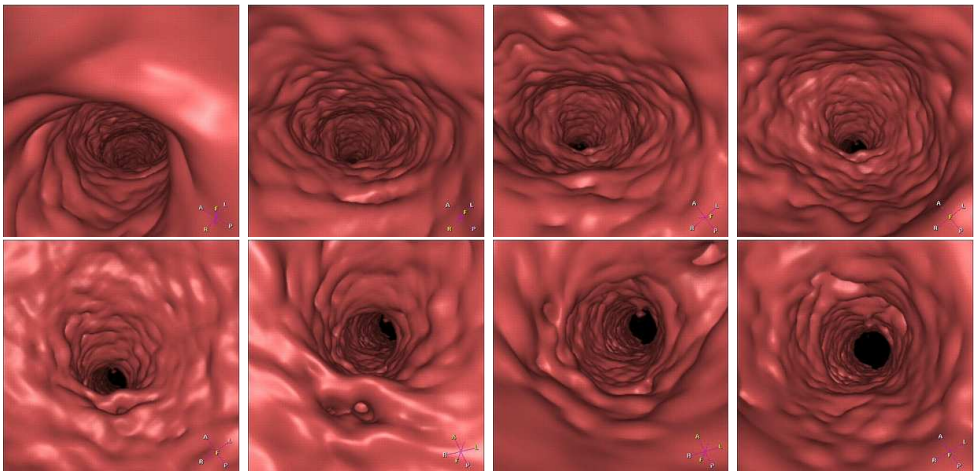


**Figure 3.14. *Virtual Endoscopy* in the Aorta:** starting from one iliac artery, the virtual camera goes to the top of the aorta, and the section of the object becomes larger when it is located inside the aneurysm.

sample of the virtual fly-through along the brain vessel is displayed in figure 3.19.

### 3.1.6 Clinical study

#### Goal

A multi-user clinical study was performed using a prototype based on EasyVision (Philips Medical Systems). The purpose was to measure the speed and user-dependence of the automatic path tracker. To this aim, the path tracking tool has been evaluated by 5 different operators :

- two physicians with manual path tracking experience ($P1$, $P2$);
- two operators with abdominal anatomy knowledge but no virtual colonoscopy practice ($M1$, $M2$);
- one reference operator ($R$) familiar with the automatic path tracking tool.

As a comparison the user $R$ also manually defined a path twice on the same dataset.

#### Data

Spiral CT data (5 mm slice thickness, 3 mm reconstruction interval) from 15 patients were used, corresponding to a total of 29 scans. During the patient preparation phase the colon was emptied as much as possible, and distended by inflating room air. In most cases, both prone and supine scanning was done.

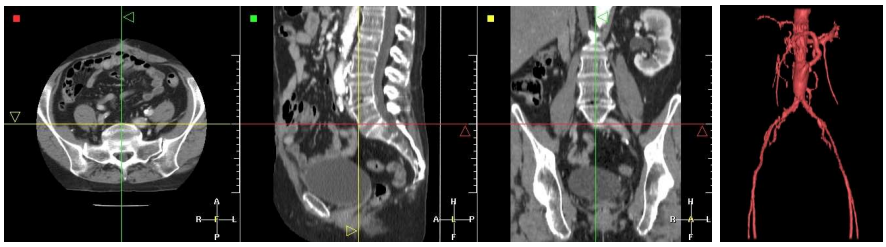


**Figure 3.15. 3D CT Scanner of the aorta:** Dataset of $512 \times 512 \times 173$ voxels; a dye product has been injected before acquisition to highlight the abdominal aorta; the anatomical object does not have a visible pathology; the right image is a threshold based volume rendering of the aorta itself (the **MIP** view is disturbed by the intensities of the bones).

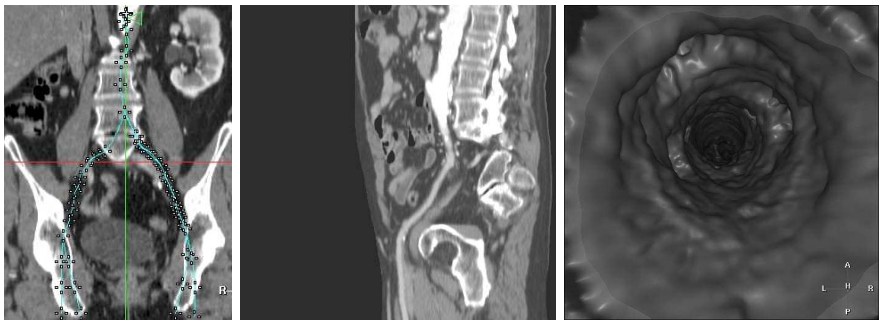


**Figure 3.16. Path extraction in the 3D CT scanner of the aorta:** Left image is the superposition of several paths extracted from the same seed point at the top of the aorta, over one slice of the dataset; middle image is a curved reformat view along one of the trajectories extracted; right image is an endoscopic view along this path.

### Measurements

**Time** For each user, the wall clock time was measured using an automatic logging mechanism. In general, path construction consisted of following steps :

1. load and inspect data
2. place starting point in the cecum
3. track + center path
4. check result and modify/continue tracking if necessary

The time necessary to perform steps 3 and 4 were measured and both user interaction time and calculation time were taken into account.

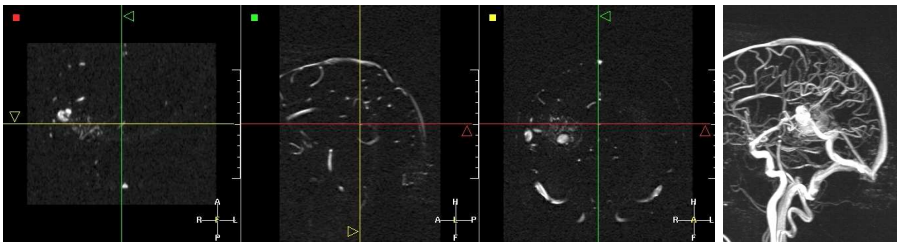


**Figure 3.17. 3D MR Angiography image of the brain vessels:** Dataset of $256 \times 256 \times 150$ voxels; It is obtained by subtracting two acquisition: one before, and one after injection of a dye product; thus there is only signal coming from moving objects; right image is a **MIP** view of this dataset.

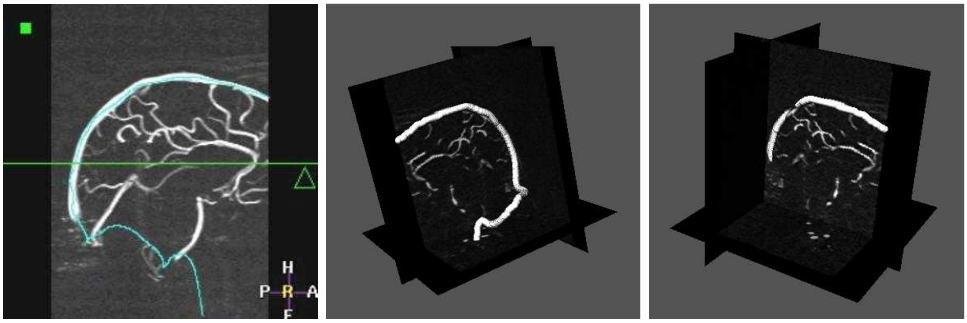



**Figure 3.18. Path extraction in the 3D MR Angiography image of the brain vessels:** Left image is the superposition of one path extracted between two points manually located inside the superior sagittal sinus over a slice of the dataset; middle and right images are the representation in 3D of this trajectory intersecting three planes, where the dataset has been mapped.

**User-dependence** To measure the user-dependence of the path tracker, resulting paths $P$ from different users were compared to the corresponding path $R$ obtained by reference user in the following way :

1. **Warp path $P(i)$ and $R(j)$ to a common length parameter $k$**
   We want to locally stretch and compress paths $P(i)$ and $R(j)$ using warping functions $w_P(k)$ and $w_R(k)$. The goal is to obtain the optimal warping satisfying

$$(w_P, w_R) = \arg \min_{w_1, w_2} \left( \sum_k \| P(w_1(k)) - R(w_2(k)) \| \right) \tag{3.1}$$

   where all points of $P$ and $R$ are addressed by the warping. This mapping is calculated using the Dynamic Time Warp algorithm [145].

2. **Calculate Euclidean distances $d$ between corresponding points**

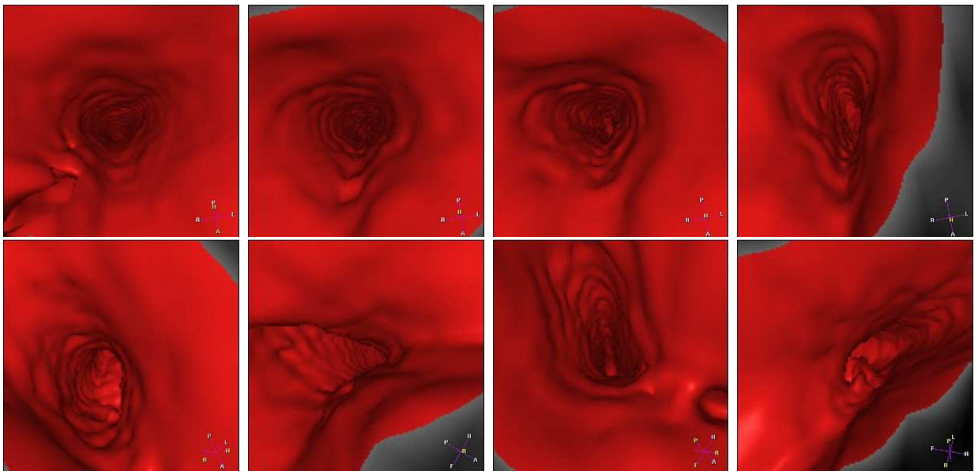


**Figure 3.19. *Virtual Endoscopy* in a brain vessel:** the virtual camera goes into the superior sagittal sinus venous canal.

After warping, the path distances $d$ can be calculated as

$$d_{P,R}(k) = \|P(w_P(k)) - R(w_R(k))\| \tag{3.2}$$

and are shown in Fig.3.20 as a function of the common path length $k$.

3. **Extract the common part**
   To exclude the effect of the exact start and end point, we only consider the common part of paths $P$ and $R$, which is defined in Fig.3.20.

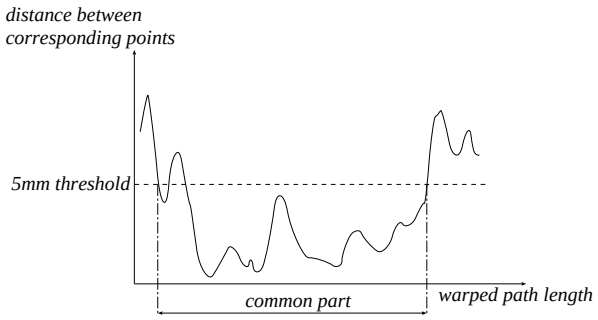



**Figure 3.20.** The Euclidean difference between corresponding points plotted versus the warped path length $k$. The common part is defined as the longest possible stretch between two positions where the distance falls below the threshold of 5 mm.

4. **Measure the similarity $\mathcal{S}$ between the paths**

We propose to measure the similarity by using the percentage of the common path length $k$ where the distance $d$ is below a threshold $t$, written as $\mathcal{S}^t_{(P,R)}$.

The chosen measure for path similarity $\mathcal{S}^t_{(P,R)}$ is symmetrical, it is robust to a complete different choice in starting position, and it gives a more reliable result of similarity between paths than e.g. the maximum distance.

### Results

**Time** The results of the time measurements are shown in Table 3.1. The average time needed for path tracking is 4.8 minutes per scan, measuring both user interaction time and calculation time.

| time | P1 | P2 | M1 | M2 | R | average | manual |
|---|---|---|---|---|---|---|---|
| **user time** | 2.8 | 4.6 | 5.3 | 4.5 | 2.0 | 3.6 | 30.0 |
| **calculation time** | 1.0 | 1.0 | 1.2 | 1.7 | 1.3 | 1.2 | |
| **total time** | 3.8 | 5.6 | 6.5 | 6.2 | 3.2 | **4.8** | 30.0 |

**Table 3.1. Timing results of the automatic path tracker**:Time expressed in minutes per scan.

No significant differences were found between the experienced physicians ($P1$ and $P2$) and the other users ($M1$, $M2$), excluding the reference user $R$. These results are in agreement with a previously published study [147], where an average of 4.5 minutes was measured on 27 cases. The timing for the manual case has no statistical meaning, but is merely given for comparison.

**User-dependence** Table 3.2 summarizes the measurements of the path correspondence using the similarity measures $\mathcal{S}^2$ (2 mm threshold) and $\mathcal{S}^5$ (5 mm threshold) as defined in section 3.1.6.

| correspondence | P1 | P2 | M1 | M2 | average | manual |
|---|---|---|---|---|---|---|
| $\mathcal{S}^2$ | 79% | 89% | 81% | 92% | 85% | 20% |
| $\mathcal{S}^5$ | 89% | 97% | 93% | 95% | **94**% | 68% |

**Table 3.2. Correspondence with reference path:** Correspondence expressed in percentage of the path length where the reference path is closer than 2 mm (first row) or 5 mm (second row).

On the average, an automatically defined path differs less than 5 mm from the reference over 94% of its length. Again, the results of the manual path tracking is given for comparison. Both manually tracked paths differed less than 5 mm over only 68% of their lengths.

Automatic path tracking provides a fast and easy way to determine the colon centerline. The resulting path lies completely inside the colon, is as much centered as possible and is smooth.

The path tracker can be used by less experienced operators without significant differences in time or resulting centerline. This is an important result, since it allows to separate the path tracking task from the actual inspection task. The former task can be done as a preprocessing step by a different person since the results of the path tracker are largely operator-independent. Separating path tracking and inspection will increase the physician's efficacy, reduce the cost and allow a more widespread application of path-based navigation and visualization.

### 3.1.7 Last developments and perspectives

Figure 3.21 shows how the colon surface is unfolded using the point of view of the virtual camera which is given by the centered path extracted. This unfolding[1] enables

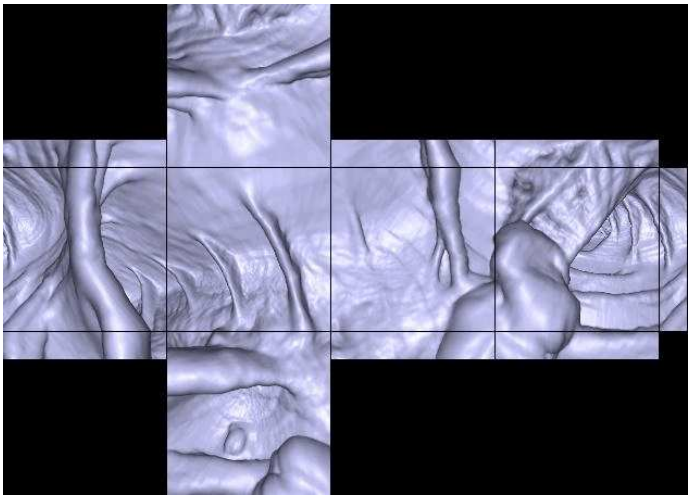


**Figure 3.21. Unfolding the colon surface:.**

to clearly see at each path position desired the rendering on each side of the camera and behind, thus increasing the accuracy of the diagnosis. For example, in the lower part of figure 3.21, the bump on the colon surface is a polyp that would not be detected if the point of view given follows the trajectory direction. Therefore, the only requirement for an accurate fly-through is to provide a trajectory that follow the centerline of the structure, as closely as possible, as this provides the best visualization of the surrounding walls. The path should also be smooth without unnecessary kinks.

Further work on the subject of endoscopy could be to simulate directly the trajectory of an endoscope inside the anatomical objects. The introduction of the angle as

[1]This image was provided by Roel Truyen, from MIMIT Advanced Development, Philips Medical Systems, Best, Netherlands.

a dimension in *Eikonal equation* in section 2.4, and the algorithmic tricks developed to compute a minimal action for a moving object in the same section, could lead to the extraction of minimal paths in the human body for moving objects with a given shape. A straightforward application of this angular propagation for an object trajectory is to guide objects in a virtual endoscopic process. If the object is not a point, the modification of its orientation will now represent a cost to minimize. If this cost is also related to the refraction indices of the medium, the constraint will regularize not the trajectory of the object, but its orientation. For a virtual endoscopic camera, regularizing the point of view will lead to a better understanding of the scene and of the pathologies. Of course, the computing time is multiplied by the number of discretized angle in $[0; 2\pi]$, thus the problem is now four dimensional, and the path extraction is really time consuming. But we can forecast that the growing computer performance in the next years will make this improvement feasible.

## 3.2 Live-Wire

Approaches in image segmentation are numerous, ranging from fully automatic methods to fully manual methods. The first ones totally avoid user's interaction but still are an unsolved problem: even if they are well adapted to specific cases their success can not be guaranteed in more general cases. The second ones are time-consuming, unrepeatable and inaccurate. To overcome these problems, interactive (or semi-automatic) methods are used. They combine knowledge of the user and computer capabilities to provide supervised segmentation, ranging from manual painting to minimal user intervention.

The target of this application was interactive and real-time extraction of features in medical images, independently from any acquisition modality. The aim was to develop a method to offer the possibility to a non-expert to draw quickly the boundary of an anatomical object. He could for example restrict its intervention to the deposition of a start point in an image. Then a contour had to be automatically found and drawn in real-time between this start point and the current cursor position. This contour should respond to a certain amount of constraints, such as internal and external forces and action of the user. The developed method should let the user validate or not the result. Taking this validation into account, the tool should be able to generate a kind of learning to better estimate the different parameters of the model.

In contour oriented segmentation, one approach is to define a boundary as the minimum of an energy function that comprises many components such as internal and external forces. In the literature, there exist many techniques to perform this minimization. First, classical active contours (also called Snakes or Deformable Boundaries), introduced by *Kass, Witkin and Terzopoulos* [82], have received a lot of attention during the past decade. But this technique presents three main problems. First, variational methods are very sensitive to the initialization step and often get trapped in a local minimum. Second, user's control cannot be applied during the extraction but only during the initialization (where the user has a whole contour to draw) and the validation stages of the segmentation. Third, the different parameters of the model are not meaningful enough in a user's viewpoint, especially for clinicians.

The application of the minimal path theory to image segmentation is a more recent technique. With this approach the image is defined as an oriented graph characterized by its cost function. The boundary segmentation problem becomes an optimal path search problem between two nodes in the graph. This approach overcomes the problem of local minima by using either dynamic programming (*Dijkstra* [43]), or a front propagation equation (*Cohen and Kimmel* [34]), mapping the non-convex cost function into a convex function. Dynamic programming has also been used for classical snakes [4], but the proposed method is applied there to find the local deformation from an initial curve that gives the best energy descent. *Falcao and Udupa* with their *Live-Wire* [48,49] and *Mortensen and Barrett* with their *Intelligent Scissors* [126–129] introduce interactivity into the optimal path approach. Their method is based on Dijkstra's graph search algorithm and gives to the user a large control over the segmentation process. The idea is the following: a start point is selected by the user on the boundary to be extracted, and an optimal path is computed and drawn in real time between this start point and the current cursor position (see figure 3.22). Thus, user's

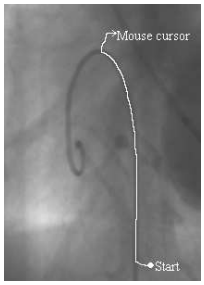



**Figure 3.22. Principle of interactive contour extraction with optimal path method:** the optimal trajectory is extracted in real time between a user defined starting point and the mouse cursor.

control is applied also during the extraction. *Mortensen and Barrett* [10,127] also introduce facilities called *Path-Cooling* and *On-The-Fly* training, which respectively lead to the possibility of drawing a closed contour, and to partial adaptation of the graph cost function. This technique seems to be the most adequate according to our target. The application consisted therefore, first in the implementation of a method based on minimal path search inspired from *Live-Wire* and *Intelligent Scissors* tools, and second in using this implementation to go further with the idea of interaction and learning. The optimal path approach is developed in the first part of this section through the description of *Live-Wire* [49] and *Intelligent Scissors* [127]. The second part explains the adaptation we made of these techniques, including adjustments and improvements.

### 3.2.1 Existing methods

The aim of this work is interactive segmentation of contours in images. In contour oriented methods, one approach is to define the boundary as the minimum of an energy function, also called cost function. This cost function includes external energy

terms describing the salient features that can be extracted from the image and internal energy terms ensuring the regularization of the segmented curve. With the minimal path approach the minimization is not global but local: the aim is to find the optimal boundary segment between two points, that is to say the contour segment where the energy is minimal. This can be achieved using the graph search theory, as detailed in section 1.3.1 .

Two different version of the minimal path extraction were used in the following:

1. first one is the *Dijkstra* algorithm [43], which is detailed in section 1.3.1.
2. second one is the *Fast-Marching* implementation presented in section 1.3.2.

Once the method is chosen, the result mainly depends on the choice for an acceptable cost function.

### Cost function

The optimal-path search is guaranteed to find the solution of the minimization of the energy function between two points. But if this function does not pertain enough to the object to extract, the contour obtained by this method won't be right. That is why defining a good and appropriate cost function is the essential task of this method. In both techniques, the processes are similar: the first step is the definition of some interesting features (F). A feature is supposed to describe certain properties of the boundary and of its environment (gray levels of the contour, of the background, ...). The next step is the conversion of these features into cost functions (C). A feature can for example be the gradient magnitude of the image, and the associated cost function can be the inverse of the gradient magnitude, giving higher cost to smaller gradients (i.e. weak contrasts) and lower cost to higher gradients (i.e. strong contrasts). The conversion of a feature into a cost function is achieved by a so called cost assignment function (CAF). Figure 3.23 illustrates all these concepts.

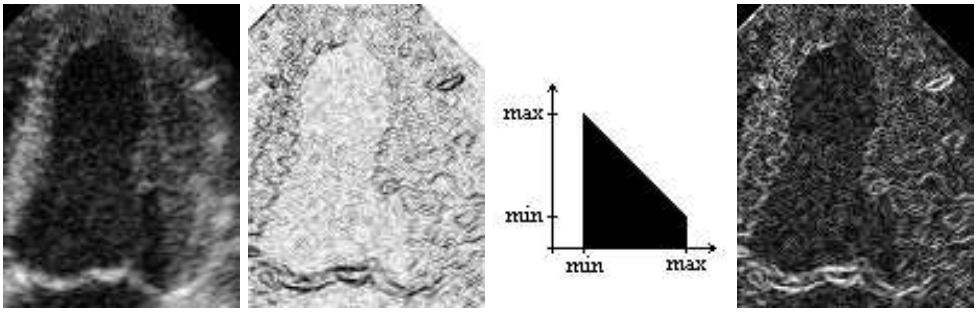



**Figure 3.23. Illustration of the cost assignment function**: From left to right: the initial image; the feature (gradient magnitude here); the *CAF* (an inversion); the new defined penalty (the inverse of the gradient magnitude).

The following list records some features quoted in the *Live-Wire* [49] and *Intelligent Scissors* [127] papers:

- Gradient magnitude,
- Direction of the gradient magnitude,
- Laplacian,
- Intensity on the positive (inside) side of the boundary,
- Intensity on the negative (outside) side of the boundary,
- Intensity on the boundary (edge).

The cost assignment process depends on how one wants to emphasis one or another value of the feature. For the gradient magnitude the inverse can for example be taken as a *CAF* in order to favor high contrasts, but a Gaussian function, centered on the gradient value one wants to highlight, could also be applied. For the Laplacian feature, the *CAF* is usually a zero-crossing detector. But many other functions may be used according to the feature values to highlight.

Once satisfying individual cost functions are available, the last step consists in combining them into a total cost function. Let us call potential the weighted sum of all the individual cost functions. This word of potential comes from the Active Contours approach where the energy of the boundary is defined as the integral along this boundary of a functional called potential. On the directed graph-arc from a pixel p to an adjacent pixel q, the potential used by *Intelligent Scissors* [127] is defined by equation:

$$\begin{aligned} \mathcal{P}(p,q) &= \omega_g \mathcal{C}_g(q) + \omega_L \mathcal{C}_L(q) + \omega_d \mathcal{C}_d(p,q) \\ &+ \omega_i \mathcal{C}_i(q) + \omega_o \mathcal{C}_o(q) + \omega_e \mathcal{C}_e(q) \end{aligned} \tag{3.3}$$

where $\mathcal{C}_x$ are the cost functions associated to the features as follows:

- $\mathcal{C}_g$: gradient feature;
- $\mathcal{C}_d$: gradient direction feature;
- $\mathcal{C}_L$: Laplacian feature;
- $\mathcal{C}_I$: inside intensity feature;
- $\mathcal{C}_O$: outside intensity feature;
- $\mathcal{C}_E$: edge intensity feature;

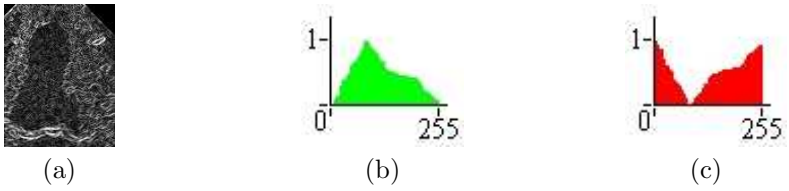
and each $\omega_x$ is the weight of the corresponding cost function.

The energy of a path is then defined by

$$E_{\text{path}} = \sum_{(p,q)\in \text{path}} \mathcal{P}(p,q) \tag{3.4}$$

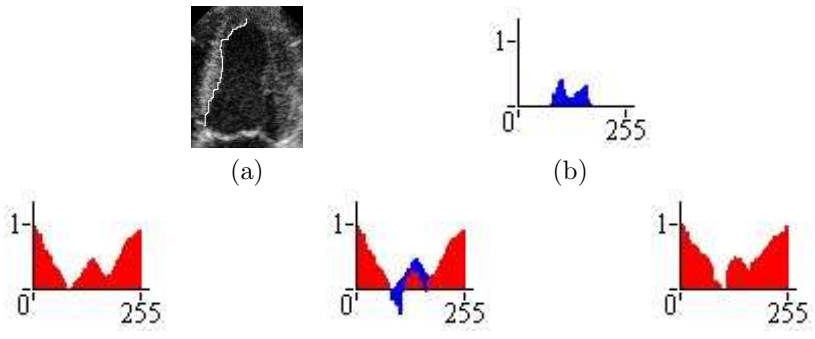
### *On-The-Fly* training

For some features it is hard to decide without prior knowledge about the boundary to extract which values are to be preferred in the cost function. The notion of *On-The-Fly* training is introduced in *Intelligent Scissors* [127] and consists in adapting, during the extraction, the cost functions of the features to the specificities of the contour one wants to segment. It is done for each feature independently from each other. The idea is the following: assuming that the user has drawn a long enough and valid boundary segment, the cost function of the feature has to be modified in order to favor contours with the same feature-values than those found on the segment. An example with the edge intensity feature is shown in table: if the extracted boundary segment is rather dark, the cost function will be modified in order to favor dark intensities. In practice, the process does not modify directly the cost function but the $CAF$ : the feature values found on the valid boundary segment (called training path) form an histogram, called training histogram, and the cost $CAF$ is iteratively modified by removing from it the training histogram and scaling the result between 0 and 1. Figure 3.24 and 3.25 illustrates respectively the initialization and an iteration of this process. The interest of *On-The-Fly* training is that the potential can be adapted during the extraction, providing the possibility of following a contour with slowly changing properties.




(a) (b) (c)

**Figure 3.24. Training initialization**: (a) the features trained; (b) its corresponding histogram at initialization; (c) its corresponding cost assignment function.




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

**Figure 3.25. Training iteration**: (a) the trained path; (b) the histogram along the trained path; (c) the cost assignment function at iteration $i$; (d) the $CAF$ minus the trained histogram; (e) the scaled $CAF$ for iteration $i + 1$.

#### *Path-Cooling*

With the minimal path approach it is impossible to extract a closed contour with only one seed point and one end point. Indeed, it would mean that two different paths could pass through a same point. To extract a close contour two seed points, at least, are needed. Furthermore, if the extracted path becomes too long, the path search method will prefer shorter paths cutting through the background: it is often necessary to fix several points to draw the expected contour. *Path-Cooling* was introduced in *Intelligent Scissors* [10], as *Bordery Cooling* and achieves automatic generation of seed points. When a new seed point is generated, the boundary segment between this new point and the previous seed point is fixed (frozen). A new start point is generated when a pixel in the contour is considered to be stable enough. The stability criterion is function of both the time spent on the active boundary segment and the path coalescence (in other terms: how many times a point has been "drawn"). For every pixel in the image two counts are considered: the time history (in milliseconds) counts how long the pixel has been included in the active boundary (boundary section that is not frozen), and the redraw history counts how many times the pixel has been redrawn by the active boundary. When the mouse moves, drawing a new optimal path, the redraw history of the active pixels is incremented while the redraw history of the non-active pixels is set to 0, and the time history of the active pixels is updated by adding to the current value the while that the boundary segment was displayed and a gradient term (to have a link with the data).

Both histories have two thresholds: a low and a high. When a pixel in the active boundary satisfies the two low thresholds it becomes a candidate point. The first candidate pixel which active boundary segment contains a pixel that satisfies the two high thresholds becomes the new seed point and the rest of the active boundary is frozen. The low thresholds have to be small in order to select candidates as close to the current free point as possible. The high thresholds have to be relatively large to freeze relatively long segments.

### 3.2.2 Adaptations and improvements

Our work is more based on the *Intelligent Scissors* than on the *Live-Wire*. In this way, we adopt the pixel based graph version: the nodes of the graph are the pixels, and the oriented arcs represent the oriented links between pixels. Our cost functions are directly inspired from the article [127] and we also use *Path-Cooling* and training. The original contribution essentially lies in the introduction of a more general path search, in the adaptation of the cooling speed, and in the way the training is achieved.

#### Path search algorithms

One of our tasks was to examine the possibility of using the path extraction detailed in section 1.1.2 in the framework of 2D-*Live-Wire* and *Intelligent Scissors*. This extraction method is based on *Cohen and Kimmel* work [34] and uses *Eikonal equation* for propagation. In our implementation we use several path search methods. We used a modified version of *Dijkstra*'s algorithm, which is the basis of the methods developed in *Live-Wire* [49] and *Intelligent Scissors* [127]. We also developed another

implementation based on the path search algorithm proposed by *Cohen and Kimmel* [34], already used for virtual endoscopy in the preceding section. We compare the results obtained with both methods.

**Modified version of Dijkstra's algorithm** On elongated objects, Dijkstra's classical algorithm might perform poorly. This is due to the minimum cumulative cost principle: as the cumulative cost is defined as a sum along the path, the path's cost is a strictly increasing function of the path's length. Thus, the longer the object is, the more likely will the extraction select shorter paths cutting through the background. See an example in figure 3.26. To overcome this problem, we can use a different expression of the cumulative cost in the Dijkstra's algorithm (see [54]), which allows longer paths, by means of introducing recursivity in the cumulative costs computation (see section 2.5 for details).

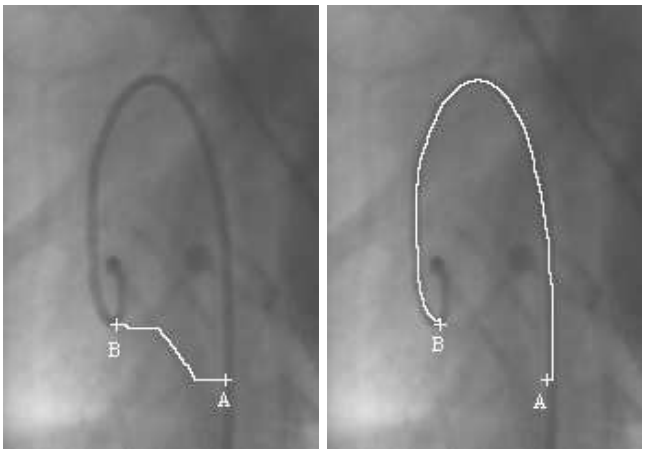



**Figure 3.26. Guide-wire extraction in a X-Ray image**: Left image - Path extraction with the classical Dijkstra's algorithm between points A and B; right image - Same path extraction with improved Dijkstra's algorithm.

**Eikonal formulation** The main idea of *Cohen and Kimmel* [34] is that the potential and the graph are considered to be continuous, producing a sub-pixel path. With this approach, detailed in chapter 1, the energy to minimize is defined as the integral of a strictly positive functional having low values close to the desired features (see equation (1.3)).

### Comparison between Dijkstra and *Eikonal equation*

As explained in the first chapter, the main difference between *Dijkstra* and *Cohen and Kimmel* definition of the minimal path lies in the considered metric. In the first case, the minimal path is the one where the sum of the potential is minimal ($L_1$ path), and in the second case, it is the one where the integration of the potential is minimal ($L_2$

path). With Dijkstra's approach, the image is considered as a graph in which each pixel is a node and the weights on the vertices are functions of the energy to minimize. This method uses dynamic programming to compute the optimal path [43]. *Cohen and Kimmel* approach keeps a continuous framework for the problem and computes the path by solving the Eikonal propagation equation in real-time with a Fast Marching algorithm. The aim of the presented work is to achieve real-time extraction. Even if Eikonal method uses integrals to compute the optimal contour, it is not very slower than Dijkstra's approach that uses sums. For the guide-wire extraction shown in figure 3.26, the computing time ration between both methods is about 0.89. And, because *Eikonal equation* produces a sub-pixel path ($L_2$ path), it is more accurate. The continuous formulation of *Cohen and Kimmel* [34] method has the advantage to keep a more general framework for the energy definition, allowing for example applications using other kind of potentials. It also easily include an offset term $w$ (see equation (1.3)) to constrain the regularity of the path, while it is more difficult with dynamic programming methods [62, 117].

### 3.2.3 Dedicated potential

To build the potential (i.e the weighted sum of cost functions), two cases are distinguished: the object to extract is either an interface between two regions, or a line (ridge) over a uniform background. The line approach was motivated by the patents [55,56], which are based on a ridge filter to proceed to the extraction of linear structures. It was therefore possible to compare quickly the two path search methods (discrete and continuous), and have a rapid overlook over the facilities of the *Live-Wire* and *Intelligent Scissors* tools. For the region interface approach, we use the six features quoted previously: the gradient magnitude, the Laplacian zero crossing, the gradient direction, the edge intensity, the inside intensity and the outside intensity, as described in [127]. Thus, the potential at an oriented arc (p,q) is described by the equation (3.3). In the next section, the defined costs functions have values scaled between 0 and 1. To display the cost maps we re-scale the values between 0 and 255 in this way: if C is the cost function and MC is the cost map associated to C, at a pixel (i,j).

#### Specific case: long curve extraction

A ridge-filter potential is used for line extraction and is based on local contrast estimation, seen as the difference between a tangential and an orthogonal term, relatively to the direction of the line to extract. It is similar to method described in [96]. Discrete and continuous implementations can be found in [60, 109]. For more details, see [54–56].

#### General case: Contour extraction

The potential used for the general case of contour extraction is the one introduced in equation (3.3).

- Gradient magnitude: this feature is useful to locate contrasted areas. As a first order derivative operator, it is a representation of the spatial variations of the image. The gradient of an image I(x,y) is defined by: $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$ and taking $G_m = \sqrt{I_x^2 + I_y^2}$ the cost function is given by

$$\mathcal{C}_g(x,y) = \frac{\max_I G_m - G_m(x,y)}{\max_I G_m - \min_I G_m} \; ; \; 0 \leq \mathcal{C}_g \leq 1$$

  The image is convolved with a Gaussian kernel, before computations.

- the Laplacian zero crossing: As a second order derivative operator, the purpose of the Laplacian zero-crossing is edge localization. The Laplacian of an image $I$ is defined by $L(I) = I_{xx} + I_{yy}$ The corresponding cost function to the Laplacian feature is the Laplacian zero-crossing ($LZC$ ). In theory it is defined like this: the $LZC$ is 0 where the Laplacian is 0 and 1 everywhere else. But in practice such a $LZC$ does not produce many zero-crossing points. That is why the zero-crossing area is extended to the pixels where the Laplacian changes its sign with a specific gap. The gap has an influence on the strength of the contrast one wants to highlight with the cost function: the deeper the gap is, the stronger the selected contrast will be. See figure 3.27. Therefore, a zero-crossing is defined

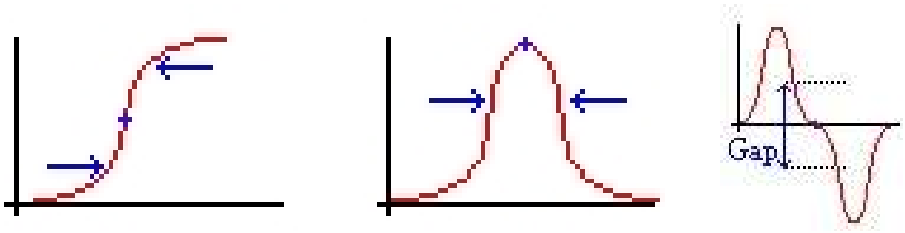



**Figure 3.27. Influence of the depth of the gap**: left image - profile of an image intensity; middle image - profile of the associated gradient; right image - profile of the associated Laplacian.

  by two points with Laplacian of opposite signs The point with its Laplacian closer to zero than the other is set to be the $LZC$ . Actually, a pixel is a $LZC$ if its Laplacian is zero or:

  - First, it is closer to zero than any of its neighbors with a Laplacian from opposite sign,
  - Then, there is at least one opposite sign neighbor so that the gap between them is sufficient.

  Consequently, the $LZC$ map is a binary map defined by:

$$\begin{cases} \mathcal{C}_L(p) = 0, \text{ if } L(p) = 0 \\ \mathcal{C}_L(p) = 0, \text{ if } \exists q \in N(p) \setminus [(L(p).L(q) < 0) \cap (|L(P)| < |L(q)|) \cap (|L(p) - L(q)| < \text{gap})] \\ \mathcal{C}_L(p) = 1 \text{ otherwise} \end{cases}$$

where $N(p)$ is the neighborhood of $p$.

- The gradient direction: introduces a smoothness constraint into the potential. The associated cost function is not local (i.e. defined on one pixel) but measures the continuity of the gradient direction between two adjacent pixels. We use the formulation as defined in *Intelligent Scissors* [127], for the gradient direction cost from pixel $p$ to pixel $q$:

$$\mathcal{C}_d(p,q) = \frac{2}{3\pi}\left\{\arccos\left[d_p(p,q)\right] + \arccos\left[d_q(p,q)\right]\right\} \ ; \ 0 \leq \mathcal{C}_d \leq 1$$

where $\arccos d_p$ and $\arccos d_q$ represent the angles between the link $(p,q)$ and the gradient direction in respectively $p$ and $q$. $d_p$ and $d_q$ are computed with

$$\begin{aligned} d_p(p,q) &= D'(p).V(p,q) \\ d_q(p,q) &= V(p,q).D'(q) \end{aligned}$$

with

$$\begin{aligned} D'(p) &= \frac{1}{\sqrt{I_x^2 + I_y^2}}(I_y(p), -I_x(p)) \\ V(p,q) &= \frac{1}{\|p-q\|}\begin{cases} q-p, \text{ if } D'(p).(q-p) \geq 0 \\ p-q, \text{ otherwise} \end{cases} \end{aligned}$$

This potential associates a low cost with an edge between two adjacent pixels where the gradient of the pixels and the link between them have similar directions. Furthermore, it associates a high cost with an edge between two adjacent pixels that have similar gradient directions but are almost perpendicular to the link between them. In practice, the efficacy of such a cost is not obvious and we do not use it in the potential. Applying a smoothing operator to the extracted curve after the extraction would probably have a better effect.

- The pixel intensities: The edge intensity is given by the scaled value of the source image at the boundary; the inside intensity is obtained at some offset k from the boundary in the gradient direction and the outside intensity comes from the image intensity at the same offset k from the boundary in the opposite of the gradient direction. Calling $\mathcal{C}_e$, $\mathcal{C}_i$ and $\mathcal{C}_o$ respectively the edge, inside and outside feature costs, their formulations are:

$$\begin{aligned} \mathcal{C}_e(p) &= \frac{1}{255}\mathcal{I}(p) \\ \mathcal{C}_i(p) &= \frac{1}{255}\mathcal{I}(p + k\frac{\nabla\mathcal{I}}{\|\nabla\mathcal{I}\|}(p)) \\ \mathcal{C}_o(p) &= \frac{1}{255}\mathcal{I}(p - k\frac{\nabla\mathcal{I}}{\|\nabla\mathcal{I}\|}(p)) \end{aligned}$$

Without training these features are not used in the potential because it is impossible to decide without prior knowledge about the contour to be extracted which values are to be preferred. The aim of training is to associate with them an adequate $CAF$ .

### 3.2.4 *Path-Cooling* improvement

We tested two different *Path-Cooling* methods. Both are based on the same idea: if a point in the active contour is stable enough, the corresponding boundary segment is frozen. The first process consists in having one counter, the redraw history, and one threshold. The other approach uses the two counters described in [127], with an adaptation: the time history is updated by multiplying (and not adding) a scaled potential-driven factor instead of a simple gradient-driven factor. The multiplication has a weighting effect which gives more influence to pixels with a low potential. At a pixel $p$ and an iteration $i$ of the cooling process, the time history $\mathcal{TH}_i$ is defined as

$$\mathcal{TH}_i(p) = \mathcal{TH}_{i-1} + t_{i-1} \times [1 - \mathcal{P}(p)]$$

where $t_i$ is the consecutive time the path was active until iteration $i$ and $\mathcal{P}$ the penalty in equation (3.3).

The time history is useful if we consider that when the user does not move (redraw history fix, but time history incremented), he wants to emphasize the already drawn contour. But, as both thresholds are to be satisfied, even if the user does move very slow, or does not move at all, the active path will freeze slowly. In practice, the definition of the thresholds is not obvious and the main difficulty lies in the interpretation of the mouse movement, in terms of speed and acceleration. We can either increase the cooling speed with the mouse cursor speed or decrease it. An argument to increase the mouse speed is the following: if the user moves fast it is because there is no real difficulty with the drawing and because he considers the extracted path as valid. But if the cooling speed is proportional to the mouse speed, and if the user has to define little areas where the contour has lots of details, he must fix a manual seed point: in these areas he must go slow and the cooling process will also go slower. An argument to decrease the cooling speed is the following: the areas where the user goes slowly are the areas which are not very well defined, where the path changes quickly its aspect... , deciding then that the slower the mouse moves the quicker the user wants the path to freeze. The problem is that if the user goes too slowly in other areas (hesitating, for example), he may fix false paths. Even if the second option, with practice and taking care to remain close to the boundary, seems to be the most adequate, no choice is totally satisfying in a general case. However, the *Path-Cooling* is a very satisfying tool to extract closed boundaries, as shown in figure 3.28.

### 3.2.5 *On-The-Fly* training improvement

In the used potential (see equation (3.3)), training can be applied to the gradient magnitude $\mathcal{C}_g$, inside $\mathcal{C}_i$, outside $\mathcal{C}_o$ and edge features costs $\mathcal{C}_e$.

#### Training path

Training, as described in [127], is based on the distribution of the feature values on a valid contour segment. The signification of valid is not obvious. We tested three approaches to define such a segment (See figure 3.29):

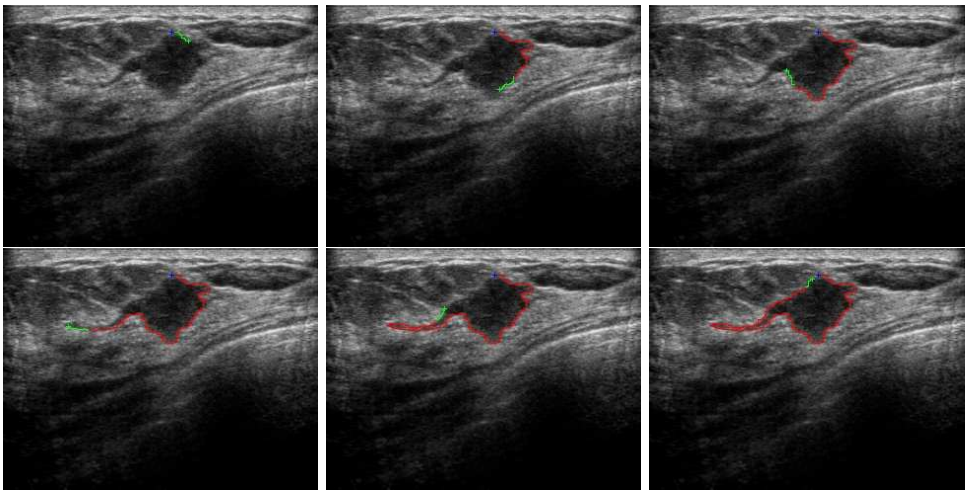


**Figure 3.28. *Path-Cooling* on an ultrasound mammography image**: These are iterations of the delineation of a tumor; the blue cross is the see point at iteration 0; the green cross is the mouse cursor; the red path is the "cooled" one; the green path is the currently drawn path between the mouse cursor and the seed point at each iteration.

1. the training path is the last section of the frozen contour and training is applied at each setting of a seed point (blue points in figure 3.29);
2. the training path is the last section of the active boundary and training is applied at each mouse movement, i.e. at each path extraction (green points in figure 3.29);
3. the training path is linked to the cursor position: it corresponds to the points where a mouse movement event is sent to the system, and training is applied at each mouse movement (red points in figure 3.29).

Training is effective when a new seed point is manually or automatically set. This setting of a seed point can be seen as the validation of a path segment. The free path, because of its high variability and dependence to the potential (and thus on the training), cannot be a suitable training area. As well as for the mouse movement marker, because it would make it impossible to go too far away from the contour with the mouse cursor.

### New way of training

We have developed an improvement of the classical training method (see patent [71]). In existing methods the training area is limited to a portion of the path itself and only takes a positive information into account (see section 3.2.1). The original technique we use is based on the addition of another training area based on negative information. The positive training area contains pixels that could belong to the contour, while

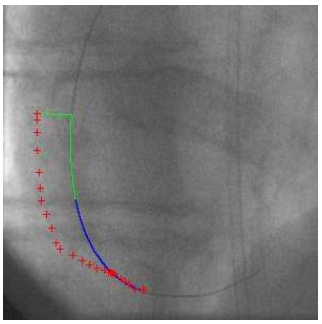


**Figure 3.29. Several training paths**: On this fluoroscopy image, the cursor position is represented by red crosses, the frozen part of the path is in blue, and the current free part of the path is in green.

the negative training area contains pixels that do not belong to the contour. The positive area has a reinforcing role while the negative area has a penalizing role in the cost assignment process. This improvement has two consequences: firstly, the new potential is more robust and secondly, the comparison of the distributions of the features (i.e. of the histograms) on each area helps adapting the weights of the individual cost functions in the total potential.

**Definition of the positive and negative training areas** The main problem is to define suitable positive and negative areas that describe well enough respectively what is and what is not a contour pixel. The definitions we use are all based on the previously described training path. As the drawing of the user is not very accurate, we consider the positive area in the neighborhood of the training path and we tested four approaches for the negative area definition:

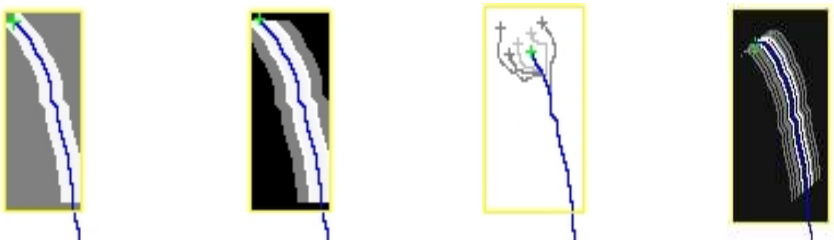
1. in the minimal box including the training path, the points closer to the path than a certain distance d are considered to be the positive area, the other points of the box are considered to be the negative area (see figure 3.30-(a));

2. in the minimal box including the training path, the points closer to the path than a certain distance dp are considered to be the positive area and the points further form the path than the distance dp and closer to the path than a certain distance dn are considered to be the negative area (see figure 3.30-(b));

3. the positive and negative areas are made from paths coming from the neighborhood of the click-position. The paths coming from the nearest neighborhood form the positive area and the path coming from the furthest neighborhood form the negative area (see figure 3.30-(c));

4. The training set of points is made from translations of the path: in the minimal box including the training path, the nearest translations are the positive area and the furthest translations are the negative area (see figure 3.30-(d));

For each approach, it is possible to add a weighting function based on the distance to the training path.



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

**Figure 3.30. Different definitions of the positive and negative areas**: Positive area in white, negative area in gray, other points of the box in black. (a) Positive area is a lane around the box, negative area is the rest of the box. (b) Positive area is a lane around the box, negative area is a lane around the positive area. (c) Areas built by the paths coming from a region around the mouse cursor. (d) Areas built with translations of the training path.

The first approach is not accurate enough for the definition of the negative area. Indeed it is possible that other points of the box belong to another right path section. It is the motivation for introducing the second approach. The third method seemed a priori to be the most accurate to define a negative area. But actually, all the paths are rapidly concurrent, what on the first hand misapprehends the notion of positive/negative training point, and on the other hand limits the size of the training set. Second and last approaches are very similar and give the best results with this difference that the first one is a continuous version of the second one. We therefore choose the last approach: the positive area is the set of p nearest translations and the negative area is the set of n next translations of the training path. The translation direction is chosen perpendicular to the mean direction of the training path. The different translations are weighted according to their distance to the training path (see figure 3.31-(d)). The negative/positive areas are symmetric for the gradient magnitude and the edge intensity features (figure 3.31-(a)). But, in order to consider the non symmetrical aspect of the inside and outside features (in equation (3.3)), we adopt for them non symmetric training areas, depending on the direction of the gradient on the path, the path direction and the considered feature. See examples with figure 3.31-(b)-(c). The positive/negative training sets of points are used to build


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

**Figure 3.31. Training areas**: In white the positive area and in black the negative one. The longest line is the training path. (a) Symmetric. (b) Asymmetric for inside feature. (c) Asymmetric for outside feature. (d) Schematic weighting function.

two distinct histograms of the features values. The function of these histograms is twofold: to build an adapted cost function for the feature (through the construction of an adapted $CAF$ ) and to adapt automatically the weight of the individual cost function in the global potential.

***On-The-Fly* adaptation of individual cost functions** Individual cost functions are built with a $CAF$ applied to the feature. Training is used to dynamically modify this cost assignment function. With our method, the algebraic difference between the positive and the negative histograms (jointly scaled) is removed from the iteratively modified $CAF$ and the result is normalized to compose a new cost.

The positive and negative histograms are scaled the following way: firstly, to have a same scale for both training histograms, the negative histogram values are multiplied by the ratio between the number of points used to build the positive histogram and the number of points used to build the negative histogram:

$$H^-[i] = \frac{\text{card}\{H^+\}}{\text{card}\{H^-\}} \times H^-[i]$$

where $H^+$ and $H^-$ are respectively positive and negative histograms. Then, both histograms are normalized using their common min/max. The initialization used in [127] favors the gray values with highest occurrence in the feature. This is incorrect initialization for images where there is a large and homogeneous background as in figure3.32-(a). We prefer an approach that does not carry any a priori about the expected feature values: the $CAF$ is initialized with a flat line, giving the same cost to each pixel of the image, and not with the inverse of the distribution of the feature. As a consequence, the CAF obtained during the process depends only on the past and the present training data.

Computing a cost function using a positive and a negative information into account makes the method more robust. Actually, the positive training area describes what is a contour and the negative training area describes what is not a contour. So if the training contour is not enough uniform, producing a too wide positive histogram, the classical approach will favor points that are perhaps not relevant to the expected contour, whereas our method the negative information to localize the contour, producing a less specific but more accurate cost function.

We show an example of *On-The-Fly* training on the septum wall of an echographic left-ventricle image in figure 3.32.

Figure 3.32 illustrates the first iteration of the process of training on the septum wall of an echographic left-ventricle image (figure 3.32-(a)) with both approaches. The trained feature is the inside intensity one (figure 3.32-(b)). The classical method uses a CAF initialized with the inverse of the distribution of the feature, where black levels are predominant and thus favored (figure 3.32-(c)). The training histogram is scaled between 0 and 1 (figure 3.32-(d)) and removed from the initial CAF to build a new CAF (figure 3.32-(e)) favoring very specific values. With this example the training path is homogeneous and the resulting cost function (figure 3.32-(f)) is good. But with a not uniform enough training contour, the resulting potential will not be consistent. Our method initializes the CAF (figure 3.32-(g)) with an arbitrary value between

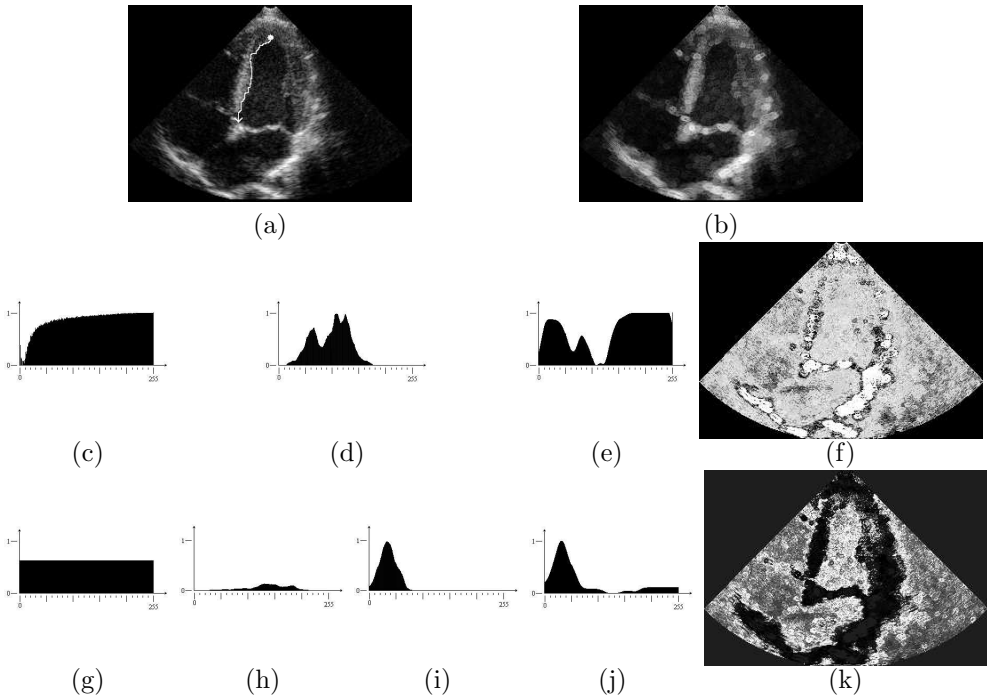



**Figure 3.32. Training on the septum wall of an echographic left-ventricle image**: (a) echographic left-ventricle image; (b) inside Feature $\mathcal{C}_i$; (c to f) **classical training**: (c) initial $CAF$ ; (d) training histogram; (e) resulting $CAF$ with (f) corresponding cost function; (g to k) **improved training**: (g) initial $CAF$ ; (h) positive and (i) negative training histograms; (j) resulting $CAF$ with (k) corresponding cost function.

0 and 1. The positive and negative histograms (figure 3.32-(h) and figure 3.32-(i)) are jointly scaled and the difference between them is removed from the initial CAF producing a less specific but carrying more accurate information histogram. Thus the new cost function (figure 3.32-(k)) is more relevant.

A real case study of the *On-The-Fly* adaptation of the individual cost functions is shown in figure 3.33, where iterations of the modification of the $CAF$ of the feature $\mathcal{C}_e$ are shown.

**Adaptation of the weights** The principal advantage of using positive and negative training areas is the evaluation of the differences between the histograms, which helps adapting the weight of the corresponding individual cost function in the global potential. If the histograms are enough distinct, we can assume that the considered feature is enough discriminating and that its weight should be more important. In a first approach we take a mean difference between both histograms as dissimilarity

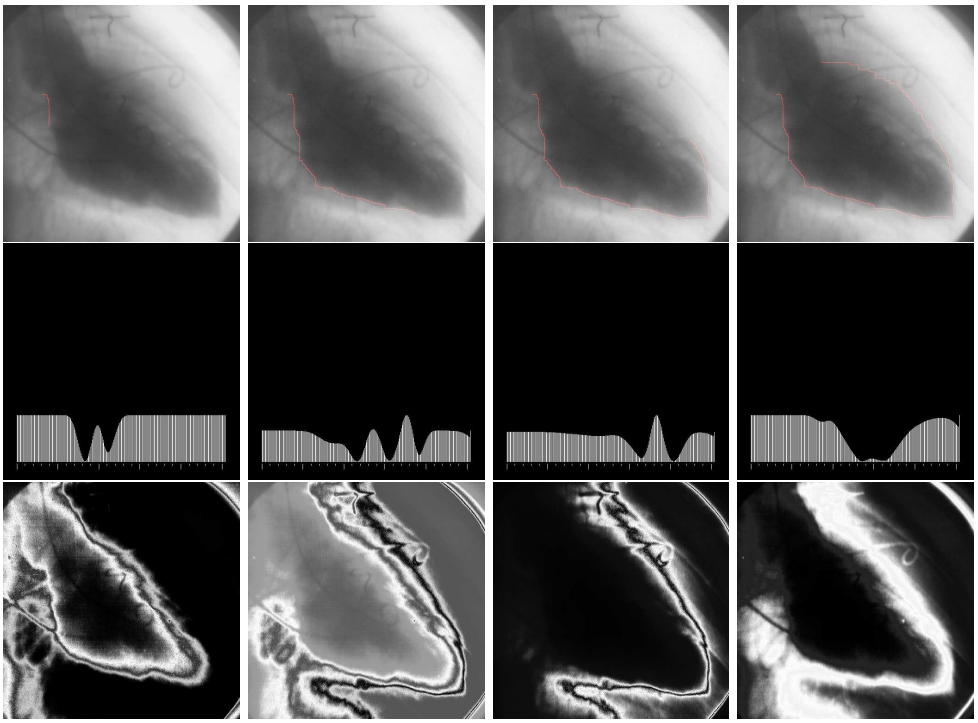


**Figure 3.33. Results of the training on a left ventricle image**: First row - iterations of the real-time contour extraction with training, on a X-Ray image of the heart left-ventricle; second row - modification of the $CAF$ of the feature $\mathcal{C}_e$ at the same iterations; third row - corresponding feature $\mathcal{C}_e$ potential at the same iterations.

criteria, which expression is the following:

$$c = \frac{1}{256} \sum_{i=0}^{255} |H^+[i] - H^-[i]|$$

Indeed, if the histograms are very similar this criterion will be very low and if they are different, it will be high. In practice, this criterion produces often very low values and quite never values above 0.5. Hence we use a stretching function to exploit efficiently this criterion and transform it into a weight $\omega_c$ associated with the cost function $c$ in the total potential. See on figure 3.34 examples of stretching functions.

### 3.2.6 Conclusion

We have developed an interactive, real-time and user-guided image segmentation software, which gives to a non-specialist the possibility to outline the contour of an object in an image without a very precise drawing (for example with the track-ball on an echograph). Figure 3.35 displays several example of the interactive drawing tool

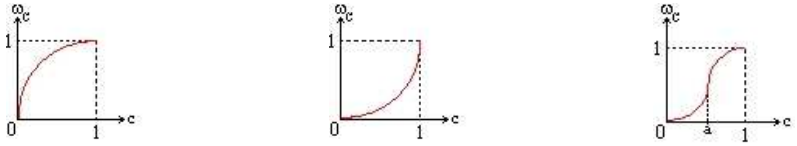



**Figure 3.34. Examples of stretching functions:** (a) Privileges small values of the cost; (b) penalizes small values of the cost; (c) penalizes values of the cost below $a$ and favors values of the cost above $a$.

for medical images. Some improvements have been brought to the bibliographical

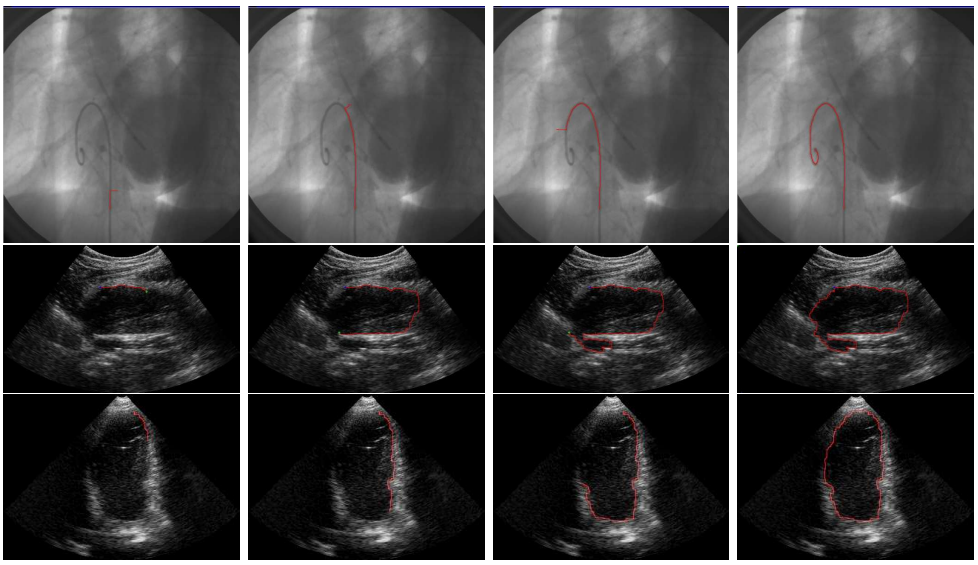


**Figure 3.35. Results on different datasets:** First row - Extraction of a guide-wire in a fluoroscopy image; second row - tumor delineation in a ultrasound mammography image; third row - extraction of the left ventricle in an ultrasound image.

background of interactive extraction of optimal path. A very general optimal path extraction method has been efficiently used, producing in real-time very precise paths. And a new method of *On-The-Fly* training has been developed to adapt, during the extraction of the contour, the individual cost-functions and their relative weights in the total potential.

### 3.2.7 Perspective

- Training:

    1. creating a better dissimilarity criterion between information from positive and negative histograms;

2. ordering the importance of each different computed criteria with the training facility, knowing at each iteration what feature is really discriminant;
3. use such a method to select *off-line* interesting features into a large database of them (like the size of the Gaussian kernel used to smooth the image).

- *Path-Cooling* : The acceleration of the mouse cursor could be an interesting information for freezing trajectories;
- 3D extension: using a 3D path search method (as in section 2.1), or perhaps a "surface-search", method instead of a slice-by-slice approach ( [46, 47]). However, with a slice-by-slice method, a contour extracted with our technique could be an interesting initialization for other pure 3D approaches (as simplex meshes [38] for example).
- Interactivity: grading the level of interactivity, and limit the user interaction to the choice of this level.

# II

# Shape Extraction


Sellier

# Chapter 4

# Deformable Models for Surface Extraction in Medical Imaging

**Résumé** — Dans ce chapitre, nous revenons sur différentes méthodes d'extraction de surface en imagerie médicale. Partant du déjà connu modèle des *snakes* de *Kass, Witkin, et Terzopoulos* [82], en section 4.1, nous présentons des méthodes basées sur une représentation explicite de la surface, comme dans les travaux de *Delingette* [38]. Nous étudions ensuite en section 4.2 une représentation implicite de la surface, à partir des travaux de *Caselles, Kimmel, et Sapiro* [22]. Nous détaillons l'implémentation de ces contours actifs géodésiques dans la section 4.3, à partir de la méthode des Ensembles de Niveaux, développée et amplement détaillée dans le livre de *Sethian* [163]. On s'intéresse en section 4.4 à définir un modèle abstrait pour nos applications, et on poursuit par son implémentation à l'aide de modèle déformables explicites puis implicites. Nous présentons par la suite l'utilisation du *Fast-Marching* comme algorithme de segmentation dans la section 4.5, et nous détaillons les applications existantes de cette méthodologie à des problèmes classiques de segmentation en imagerie médicale 3D à la section 4.6.

**Abstract** — In this chapter, we recall the different methods used for surface extraction in medical imaging. Starting from the already studied *snakes* framework of *Kass, Witkin, and Terzopoulos* [82] in section 4.1, we mention methods based on explicit representations of the shape, as done by *Delingette* [38]. Thus we extend in section 4.2 to implicit representations of the shape, proposed by *Caselles, Kimmel, and Sapiro* [22]. We detail implementation of those Eulerian active contours in section 4.3, using the level-sets methodology developed extensively by *Sethian* [163]. We develop an abstract deformable model for our applications in section 4.4, and we follow by implementations of this framework with explicit and implicit deformable models. We further detail the use of the *Fast-Marching* method to segmentation tasks in section 4.5. And we detail several applications of this methodology to medical image segmentation problems in section 4.6.

## 4.1 Classical Active Contours

### 4.1.1 Definition

We recall the *Snake* model as introduced in [82], and already mentioned in chapter 1. For a general overview on deformable models, see [115]. See also a description and references in [30].

The classical energy of the model which will be minimized on $\mathcal{A}$ the space of all admissible curves, and has the following form:

$$\begin{aligned} E: \quad & \mathcal{A} \to \mathbb{R} \\ & \mathcal{C} \mapsto E(\mathcal{C}) \quad = \int_\Omega \frac{w_1}{2}\|\mathcal{C}'(v)\|^2 + \frac{w_2}{2}\|\mathcal{C}''(v)\|^2 + P(\mathcal{C}(v))dv \end{aligned}$$

where $\Omega = [0,1]$ is the parameterization interval. This model is used in the classical formulations of deformable models [32, 102].

In this formulation, each term appears as a potential acting on the shape. Thus the mechanical properties of the deformable model are controlled by two kinds of constraints:

- The internal potential: $C'$ and $C''$ are the smoothing terms on the curve. They enable to control its regularity by means of $w_1$ which quantifies its rigidity and $w_2$ its elasticity;
- The external potential term $P$ represents the likelihood. This "image" potential traps the curve towards the regions with desired attributes.

Figure 4.1[1] displays iterations of the heart segmentation in a 3D ultrasound image of the heart, with simplex meshes.

### 4.1.2 Drawbacks

The main drawbacks of the classical deformable model approach are:

- Minimization: The functional is non-convex, and one difficulty is to find a good local minimum. Spurious edges generated by noise may stop the evolution of the surface, giving an insignificant local minimum of the energy;
- Initialization: The user must specify an initial shape that is close to the goal, like a very precise polygon approximation, which may be tedious to draw;
- Topology changes: this method is unable to segment several objects simultaneously, and to merge different shapes;

One of the main issue in using deformable models is their initialization and minimization. The solution introduced in [163] allows to solve the global minimization of a problem. His approach considers a slightly modified problem: a curve is considered as an interface between two media, following a particular evolution equation. We will see that under precise assumptions, this front evolution efficiently builds a path between two fixed points, as detailed in [34].

[1] Slice of a 3D ultrasound dataset acquired with a multi-plane trans-oesophagus scan-head on a HDI 5000 ATL echograph.

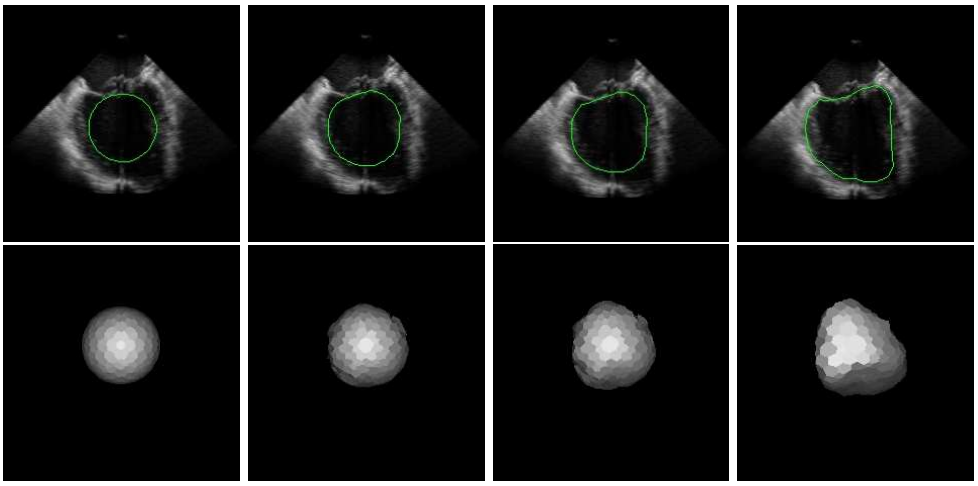


**Figure 4.1. Samples of the segmentation of a heart in a 3D ultrasound image with simplex meshes [37] :** Starting with a sphere interpolated with simplex mesh faces, we segment a heart in a 3D ultrasound image: first row shows the intersection of the mesh with a slice of the dataset, and second row is the 3D model at the same iterations.

## 4.2 Geodesic Active Contours

A geometric approach for deformable models was introduced by *Caselles, Catté and Dibos* [21] and *Malladi, Sethian and Vemuri* [113]. The basic idea of the geometric model is that the curve follows an evolution by expansion in the normal direction, with lower speed when the image force $P(\mathcal{C})$ is small. Hence the evolution of a planar curve $C$ is in the direction of its normal only is given by

$$\frac{\partial \mathcal{C}(s,\tau)}{\partial \tau} = P(\mathcal{C})(\frac{\partial^2 \mathcal{C}}{\partial s^2} + w\ \mathbf{n}) = P(\mathcal{C})(\kappa + w)\mathbf{n}, \tag{4.1}$$

where $s$ is the arc-length parameter of the curve $\mathcal{C}$, $\kappa$ is the curvature, $\mathbf{n}$ is the unit normal. The constant term $w$ is similar to the balloon force introduced in the *snakes* model [29] (and also related to the dilatation transform in mathematical morphology and the grass-fire transform [102]).

It was shown that the geometric snakes model handles topology changes better than the classical snakes when implemented with the level set approach for curve evolution proposed by *Osher and Sethian* [135, 158].

But, due to the formulation of the image force, it never comes to a complete stop, and heuristic stopping procedures are used to switch off the evolution process when an edge is reached. In equation (4.1), the geometric snake evolution is slower when the $P$ is small but the curve does not necessary stop completely at the boundary, since it never reaches an equilibrium.

Given an initial curve $\mathcal{C}(s,0)$, the *geodesic active contours* is based on the planar

evolution equation

$$\frac{\partial \mathcal{C}(s,\tau)}{\partial \tau} = (P(\mathcal{C})(\kappa + w) - \langle \nabla P, \mathbf{n} \rangle)\, \mathbf{n}, \tag{4.2}$$

where $s$ is the arclength. The $\nabla P$ term added, in comparison to the geometric model, is a projection of the attraction force $-\nabla P$ on the normal to the curve. This force balances the other term close to the boundary and causes the curve to stop there.

This introduction of $\nabla P$, based on geometrical as well as energy minimization reasoning, leads to the "geodesic active contour" proposed by *Caselles, Kimmel and Sapiro* [23]. The geodesic active contours enjoy the advantages of classical as well as geometric active contours, since it handles topology changes and reaches an equilibrium which is similar to the classical snakes.

The curve evolution equation is then reformulated and implemented using the *Osher-Sethian* numerical algorithm [135]. Similar geometric models were also introduced in [85, 149, 165, 183] and extended to color and texture in [153].

## 4.3 *Level-Sets* Implementation of the Geodesic Active Contours

Let $C_o(p)$ be a closed initial parameterized planar curve in an Euclidean plane, and $C(p,t)$ the family of curves generated by the movement of $C_0(p)$ in the direction of its outward Euclidean normal vector $\mathbf{n}$. The speed of this movement is supposed to be a scalar function of the curvature $\kappa$:

$$\begin{cases} \frac{\partial C}{\partial t}(p) &= F(\kappa)\mathbf{n} \\ C(p,0) &= C_0(p) \end{cases} \tag{4.3}$$

A Lagrangian approach might be considered to implement the curve evolution according to the above equations in motion equations of the discretized positions of $C(p)$.

But this formulation has several drawbacks. When tracking the motion of the interface $C$ propagating along its normal direction with velocity $F$, most numerical techniques rely on markers, breaking it up into points connected by segments, and moving each point with speed $F$ (see figure 4.2-left). The solution is supposed to gain in accuracy if the number of points is increased. Problems arise if different parts of the front cross each other, or if the shape tries to break into two pieces (see figure 4.2-middle and the book of *Sethian* [163] for details) or if two shapes try to merge into one (see figure 4.2-right).

Therefore, the main drawback of this approach is that the evolving model is not capable to deal with topological changes of the moving front, and external procedures must be added to detect and deal with merging and splittings. These methods were developed for active contour models, namely *Topological Snakes* (T-snakes) in [114], for triangulated meshes in [97], and for their dual simplex meshes in [121]. There is no suitable difference approximation scheme for the Lagrangian implementation, due to the time-dependent parameterization of the curve model, thus leading to stability problems.

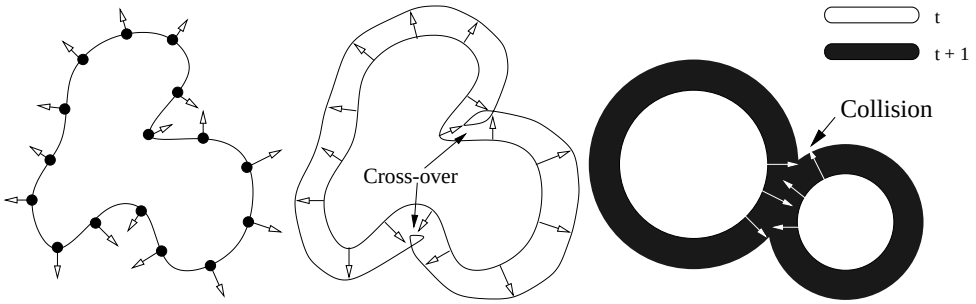



**Figure 4.2. Markers methods and related problems:** Left image illustrates the markers technique. Middle image shows the cross over which occur frequently. Right image shows problems of merging two contours.

These problems are handled very elegantly by the level set methodology, originally introduced by *Osher and Sethian* in [135], and now widely used in lots of applications, ranging from computer vision problems, to semiconductor manufacturing, shape from shading, and robotic navigation (see [163]). They embed the initial position of the moving interface $C_0(\mathbf{x})$ as the zero level set of a higher dimensional function $\phi$ (the signed distance to $C_0$, as shown in figure 4.3), and link the evolution of this new function $\phi$ to the evolution of the interface itself through a time-dependent initial value problem. At each time $t$, the contour $C(t)$ is given by the zero level-set of $\phi$.

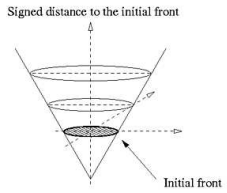



**Figure 4.3. Embedding the contour in the signed distance:** In this 2D example, the initial contour $C_0(x, y)$ is a circle, and $\phi(x, y, 0) = \pm d(C_0((x, y))$.

This condition states that

$$\phi(C(t), t) = 0 \Rightarrow \phi_t + \nabla\phi(C(t), t) \cdot \frac{\partial C}{\partial t} = 0 \tag{4.4}$$

Since $\frac{\partial C}{\partial t} = F\mathbf{n}$ and the outward normal vector is given by $\frac{\nabla\phi}{|\nabla\phi|}$, this yields the following evolution equation for $\phi$ given in [135]:

$$\begin{cases} \phi_t & + & F|\nabla\phi| = 0 \\ \phi(\mathbf{x}, 0) & = & C_0(\mathbf{x}) \end{cases} \tag{4.5}$$

### 4.3.1 Advantages of this formulation

There are several advantages associated with the *Level-Sets* paradigm:

1. This formulation remains unchanged in higher dimensions, as well for 2D curves, as for hypersurface in three dimensions (and higher). Therefore the embedded hypersurface will be denoted $\Gamma$ in the following.

2. The evolving function $\phi$ remains a functions as long as $F$ is smooth. As a consequence, topological changes in the evolving front $\Gamma(\mathbf{x}, t)$ are handled naturally, the position of the front at time $t$ is given by the zero level-set $\phi(\mathbf{x}, t) = 0$ of the evolving function $\phi$. $\Gamma(\mathbf{x}, t)$ can be several initial curves and it can break and merge as $t$ advances. A 2D examples of fronts merging is shown in figure 4.4.

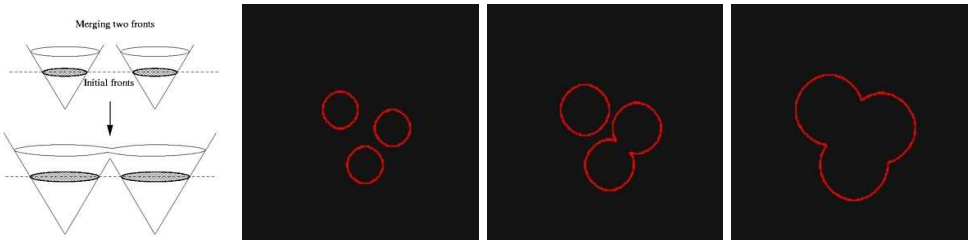



**Figure 4.4. Easy handling of contour merging with *Level-Sets* :** $\Gamma(x, y, t = 0)$ is the representation of the three curves. Initializing $\phi$ by the distance to these circles, and propagating $\phi$ with a constant positive speed in the outward normal direction, $\phi_t + \beta|\nabla\phi| = 0$, the three circles increase and merge. The different images represent the zero level-set of $\phi$ at several successive iterations.

3. Intrinsic geometry properties of the front are easily determined from the front itself, like the normal to the front $\frac{\nabla\phi}{|\nabla\phi|}$, and the curvature of the front $\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$.

4. The evolution equation (4.5) can be approximated by efficient computational schemes with finite different approximation for the spatial and temporal derivatives. Explicit finite difference approach is possible, and other implicit and semi-implicit approaches have been already developed [65]

Explicit representation of the surfaces can also handle topology changes. Several attempt to achieve this tasks were developed independently.

- *McInerney and Terzopoulos* [114,116] propose the *"T-snakes"* and *"T-surfaces"* with adaptive topology; the initial model is a triangulation that evolved according to a Lagrangian evolution equation, and the triangulation is resampled by computing its intersection with a tetrahedral griding of the image domain. By defining an "inside" and an "outside" region, the resampling handles topology changes when the surface self-intersect; this ad-hoc approach is limited to closed contours and surfaces.

- *Lachaud et al.* [97,98] propose a very interesting technique based on the distance between each vertices of the triangulation. But knowing this distance between each pair of nodes is a huge computing task;

- *Montagnat and Delingette* [39, 40, 121] propose topological changes based on the simplex mesh surface representation [37]. This method that resamples the surface on a regular grid of the image domain works well in 2D and works with low computation times, but no 3D extension is available for the moment.

The different methods mentioned are all proposing methods to adapt the topology of their objects. This method has advantage of reducing the importance of the *a priori* on the final shape of the target of the segmentation process. Therefore, the topology of the model at initialization do not need to be the same than the model at convergence, thus reducing user interaction. But these methods are based on approximations, while the *Level-Sets* formalism handles naturally this problem. The *Level-Sets* model is evolved, and its zero level-set can be represented at several iterations, as shown in figure 4.4. But this implicit representation has one major drawback: it severely limits the possible interactivity of the user on the model, as mentioned in [120]. However, we will see in the next chapter that some basic interactions can be applied to this formalism.

### 4.3.2 Different Motions of the interface

The evolution of the segmentation is performed through the evolution of $\phi$, which is done by a flow equation:

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = \frac{\partial \phi}{\partial t} + \mathbf{V} \cdot \nabla \phi = 0 \tag{4.6}$$

if we consider the vector flow field $\mathbf{V}(\mathbf{x}, t)$ with $(\mathbf{x}, t) \in \Omega \times [0, +\infty[$, which evolves $\phi(\cdot, t)$.

#### Scalar flow

Any flow of the form:

$$\mathbf{V} = \beta(\mathbf{x}, t)\mathbf{n} \text{ with } \beta(\mathbf{x}, t) \in \mathbb{R} \tag{4.7}$$

will be referred as a scalar flow. Evolution under (4.6) with positive (resp. negative) values for $\beta$ yields to a normal dilatation (resp. shrinkage) of $\phi(0, t)$. Figure 4.8 displays iterations of a front, initialized with the distance to a circle, evolving under an inflating term. In figure 4.5, $\phi$ is evolved with $\beta = -1$ in the expression of the scalar vector field of equation (4.7). Scalar flows lead to non-linear flow equations.

#### Vector Flow

Any flow of the form:

$$\mathbf{V} = \mathbf{U}(\mathbf{x}, t) \text{ with } \mathbf{U}(\mathbf{x}, t) \in \mathbb{R}^{\mathrm{d}} \tag{4.8}$$

and no dependency on $\phi$ will be referred to as a vector flow, or vector flow field. Evolution under (4.6) corresponds to passive advection of the level sets of $\phi(\cdot, t)$. Figure 4.6 displays iterations of a front, initialized with the distance to a circle, evolving under the influence of an advection flow only with $U(\mathbf{x}, y, t) = \mathbf{u} = (1, 1)$. The resulting contour is implicitly moved, in the direction of the vector $\mathbf{u}$. Vector flows lead to linear flow equations.

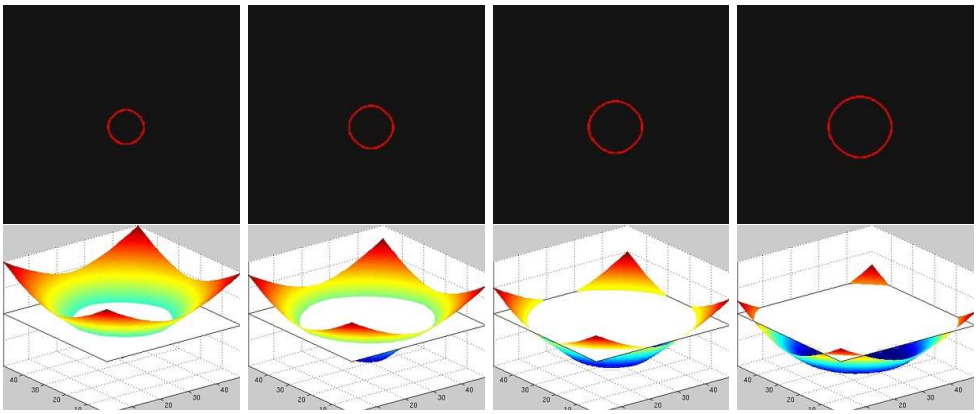


**Figure 4.5. Inflating force**: First row images are consecutive iterations of the *Level-Sets* evolution under external forces only; The external forces are built using the gradient of a left-ventricle image; Bottom row shows the zero level-set superimposed on the gradient magnitude at the same iterations.

### Curvature motion

Any flow of the form:

$$\mathbf{V} = -\varepsilon(\mathbf{x}, t)\, \kappa_M(\mathbf{x}, t)\, \mathbf{n} \text{ with } \varepsilon(\mathbf{x}, t) \in \mathbb{R} \tag{4.9}$$

will be referred to as a curvature flow. Evolution under equation (4.6) with positive values for $\varepsilon$ yields to a local regularization of $\phi(0, t)$. Negative values for $\varepsilon$ lead to instabilities. Figure 4.7 displays iterations of a front, initialized with the signed distance to an initial curve, evolving under the influence of its curvature only, using the level set equation with a speed function of the form $F(\kappa) = -\kappa$

$$\phi_t = \kappa |\nabla \phi| = \left[ \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \right] |\nabla \phi| \tag{4.10}$$

If we let the iterations proceed, the zero level-set (in black) will tend to a circle, and shrink to a point before vanishing. Equation (4.10) applied to this image illustrates Grayson's theorem that all simple closed curves moving under its curvature must shrink to a point (as shown in [70]), regardless of its initial shape. This motion resembles a non-linear heat equation (see [163]) and smoothes large oscillations, and can be used in curve extraction as a diffusion term, to relax boundaries. Curvature flows lead to non-linear flow equations.

### Composite flow

Any flow which is the sum of flows of the preceding types will be referred to as a composite flow. Figure 4.8 displays iterations of a front, initialized with the distance to a circle, evolving under the following equation:

$$\phi_t + g_I(1 - \varepsilon\kappa)|\nabla \phi| - \beta \nabla P \cdot \nabla \phi \tag{4.11}$$

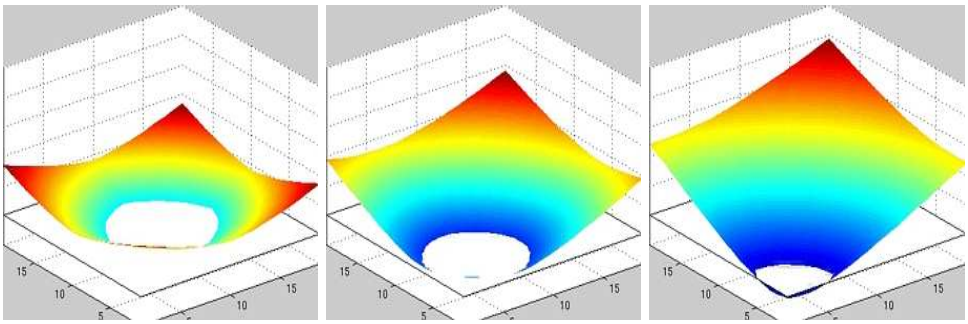


**Figure 4.6. Advection flow**: the level sets are represented using a colored ladder; the zero level-set is implicitly defined by the intersection between the levels and the white plan at height zero.

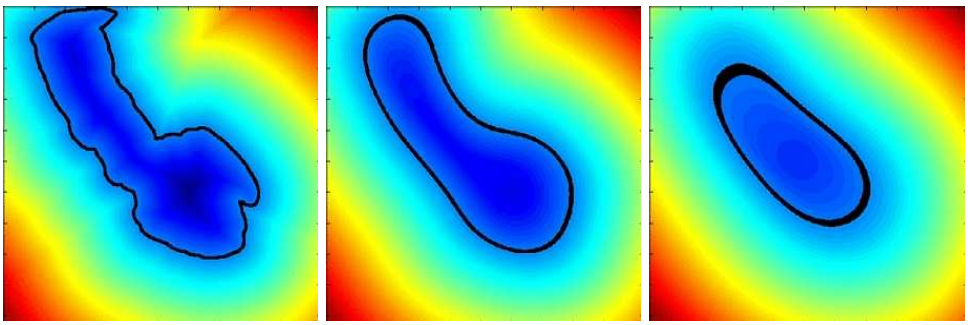


**Figure 4.7. Curvature motion**: the zero level-set (in black) and all the other levels evolve according to their curvature.

where the equation contains the following terms:

1. an inflating force, as in equation (4.7), here defined by

$$g_I(\mathbf{x}) = \frac{1}{1 + |\nabla I_\sigma(\mathbf{x})|} \tag{4.12}$$

   where $I_\sigma$ is the image convolved with a Gaussian kernel of size $\sigma$; this inflating force vanishes to zero in region of high gradients (i.e. near object boundaries)

2. a curvature term $-g_I \varepsilon \kappa$ which controls the smoothness of the iso-contours of $\phi(\cdot, t)$, originally introduced in [112];

3. an external force, which role is to attract the surface towards the boundary of the object of interest. This term is a vector flow which denotes a projection of an attractive force vector on the surface normal, in our case we use a potential field defined as in [22], by

$$P(\mathbf{x}) = -|\nabla I_\sigma(\mathbf{x})| \tag{4.13}$$

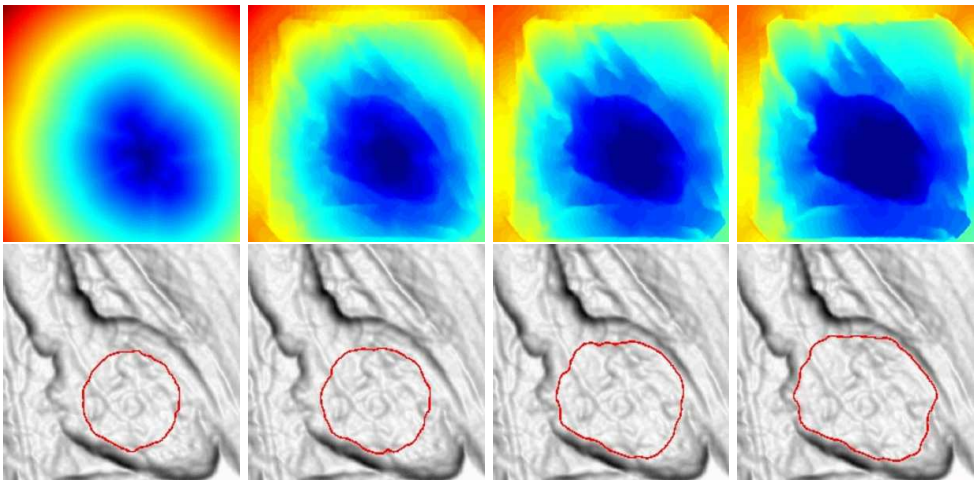


**Figure 4.8. Composite flow**: First row images are consecutive iterations of the *Level-Sets* evolution under the equation (4.11); the external forces are built using the gradient of a left-ventricle image; bottom row shows the zero level-set superimposed on the gradient magnitude at the same iterations.

## 4.4 Region-based forces

In this chapter we present a variational framework for the segmentation on the basis of the geodesic contour implementation of region-based forces. This framework has been used in the following parts of the thesis.

### 4.4.1 Partitioning the image domain

Region-based terms have already been included in active contour models [24, 25, 27, 148] We consider an *abstract deformable model*, which consists of a partition of the *image domain* $\Omega$ into three subsets: two open subsets $\Omega_{in}(t)$ and $\Omega_{out}(t)$ (the *inside* and the *outside* of the segmented object) and their common boundary $\Gamma(t)$. $\Gamma(t)$ is supposed to be as smooth as necessary (for example we may suppose that $\Gamma(t)$ is a locally Lipschitz-continuous hypersurface). This partition is a function of an evolution parameter $t \in [0, +\infty[$ that will be called *time*[2]:

$$\begin{cases} \Omega = \Omega_{in}(t) \cup \Gamma(t) \cup \Omega_{out}(t) \\ \Gamma(t) = \partial\Omega_{in}(t) = \partial\Omega_{out}(t) \end{cases} \tag{4.14}$$

Figure 4.9 shows an example of 2D image, with the abstract deformable model as we would like it to look like after segmenting the image.

[2]this "artificial" time has nothing to do with the notion of physical time.

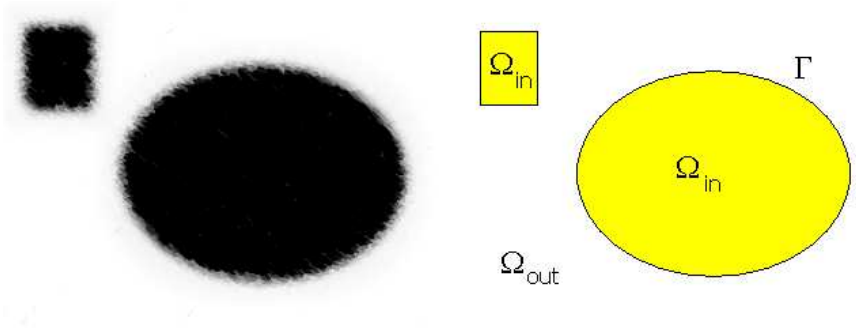



**Figure 4.9. Abstract Deformable Model:** left image is the 2D synthetic data; right image is the 2D abstract deformable model.

### 4.4.2 Region descriptors

Region descriptors are real functions of the image intensity $I$, and of $\Omega_{in}$ and $\Omega_{out}$. We consider:

$$\begin{cases} k_{in}(\mathbf{x},t) = -\log P_{in}(\mathbf{x},t) \\ k_{out}(\mathbf{x},t) = -\log P_{out}(\mathbf{x},t) \end{cases} \tag{4.15}$$

where $P_{in}(\mathbf{x},t)$ (resp. $P_{out}(\mathbf{x},t)$) is the probability that $x$ is in $\Omega_{in}(t)$ (resp. $\Omega_{out}(t)$).

Gaussian descriptors were first introduced by *Zhu and Yuille* [196]. They are "fully automatic" in the sense that no user-defined parameter is necessary to their definition. They are based on region probabilities defined by Gaussian distributions:

$$P_{in}(\mathbf{x},t) = \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{\left(I(\mathbf{x})-\mu_{in}(t)\right)^2}{2\,\sigma_{in}^2(t)}) \tag{4.16}$$

A similar expression is used for the definition of $P_{out}(\mathbf{x},t)$. Here $\mu_{in}(t)$ and $\sigma_{in}^2(t)$ are respectively the mean and the variance of the image intensity over $\Omega_{in}(t)$:

$$\mu_{in}(t) = \frac{\int_{\Omega_{in}(t)} I(\mathbf{x})\,d\mathbf{x}}{\int_{\Omega_{in}(t)}\,d\mathbf{x}}\ ,\ \sigma_{in}^2(t) = \frac{\int_{\Omega_{in}(t)}(I(\mathbf{x})-\mu_{in}(t))^2\,d\mathbf{x}}{\int_{\Omega_{in}(t)}\,d\mathbf{x}}$$

This means that the histograms of the intensity in $\Omega_{in}(t)$ and $\Omega_{out}(t)$ are *modeled* by Gaussian distributions. Those descriptors were also used extensively when *Paragios* introduced the concept of *Geodesic Active Regions* [137], for unsupervised image segmentation, supervised texture segmentation [139], and detection and tracking of moving objects [138].

### 4.4.3 Boundary descriptors

The boundary descriptor $k_b$, which is sometimes called *edge indicator*, is usually a real function of $\nabla I$. We took the classical expression inherited from geodesic active

contours (see for example *Caselles et al.* [23] or *Paragios* [137]):

$$k_b(\mathbf{x}) = \frac{1}{1 + \frac{\|\nabla I(\mathbf{x})\|^2}{\gamma^2}}$$

where $\gamma$ is a user-defined parameter.

To make this descriptor "fully automatic", it is possible to use the mean gradient magnitude in the entire image and set:

$$\gamma = \frac{\int_{\Omega(t)} \|\nabla I(x)\| \, d\mathbf{x}}{\int_{\Omega(t)} d\mathbf{x}}.$$

This choice gave very good results on almost all the images we segmented.

Among other possibilities for boundary-based descriptors, we must mention the excellent work of *Xu and Prince* [189–191]. The construction of a new gradient information, using a somehow anisotropic diffusion technique to extend the local gradient information in the whole image, has been recently used together with the *Level-Sets* formalism in [140].

### 4.4.4 Segmentation as an optimization process

#### Composite energy functional

We see the process of segmentation of the image as the minimization of an *energy functional*, also called *objective function*. We decided to use a composite energy functional which comprises *region-based* and *boundary-based* terms. We took:

$$J(t) = \zeta \int_{\Omega_{in}(t)} k_{in}(\mathbf{x}, t) \, d\mathbf{x} + \zeta \int_{\Omega_{out}(t)} k_{out}(\mathbf{x}, t) \, d\mathbf{x} + \eta \int_{\Gamma(t)} k_b(\mathbf{x}) \, d\sigma \qquad (4.17)$$

where $d\mathbf{x}$, $d\sigma$, $\zeta$ and $\eta$ are respectively the Lebesgue measures of $\mathbb{R}^{\mathrm{d}}$ and $\Gamma(t)$, and two user-defined positive scalar parameters.

The integrands in 4.17 are called *descriptors*, and contain all of the information that is being extracted from the image. The functions $k_{in}$ and $k_{out}$ are the *region descriptors*, and $k_b$ is the *boundary descriptor*. The punctual values of the region descriptor $k_{in}$ (resp. $k_{out}$) are supposed to be all the smaller as the probability to be in the interior (resp. exterior) of the object to be segmented is high. Similarly, the punctual values of the boundary descriptor are supposed to be all the smaller as the probability to be near a physical contour in the image is high.

The scalar parameters $\zeta$ and $\eta$ may be adjusted to give more weight to the region-based integrals or the boundary-based integrals. The choice of $\zeta$ and $\eta$ depends on the application specificities (quality and contrast of the images, image scale[3], etc.) and of the confidence of the user in the different descriptors.

[3] Gaussian pre-smoothing of the image.

**Minimization of the energy functional**

The evolution of the partition is totally defined by the evolution of the interface $\Gamma$ between $\Omega_{in}(t)$ and $\Omega_{out}(t)$. The evolution (or motion) of $\Gamma$ is defined by a speed field $\mathbf{V}(p,t)$, where $p$ is the parameter corresponding to the chosen parameterization of $\Gamma$:

$$\frac{\partial \Gamma(p,t)}{\partial t} = \mathbf{V}(p,t). \tag{4.18}$$

We recall that the evolution of a hypersurface under an intrinsic (i.e. independent of the parameterization of the hypersurface) speed field depends only on the normal component of the speed (see Keriven [84, p. 40]).

The minimization of $J$ is done by a gradient descent defined by the evolution equation (4.18). Several proofs have been proposed for the derivation of $J$. The most correct approach can be found in [79]. It shows that the evolution equation (4.18) implies that the temporal derivative of $J$ is given by:

$$\begin{aligned}\frac{dJ}{dt} = \quad & \zeta \int_{\Omega_{in}(t)} \frac{\partial k_{in}}{\partial t}\, d\mathbf{x} + \zeta \int_{\Omega_{out}(t)} \frac{\partial k_{out}}{\partial t}\, d\mathbf{x} + \\ & \int_{\Gamma(t)} (\zeta\,(k_{in} - k_{out}) + \eta\,(k_b\,\kappa_M(p,t) + \nabla k_b))(\mathbf{V}.\,\mathbf{n})\, d\sigma\end{aligned}$$

where $\mathbf{n}$ and $\kappa_M$ are respectively the normal (directed from $\Omega_{in}(t)$ to $\Omega_{in}(t)$) and the mean curvature of the hypersurface $\Gamma(t)$. In the case of time-independent region descriptors, the speed that minimizes the energy functional is consequently given by:

$$\mathbf{V} = \zeta\,(k_{out} - k_{in})\,\mathbf{n} - \eta\,(k_b\,\kappa_M(\mathbf{x},t)\,\mathbf{n} + \nabla k_b). \tag{4.19}$$

The evolution equation (4.18) will be embedded into the *Level-Sets* formulation described in section 4.3, and the flow (4.19) will be implemented using the different flows detailed in section 4.3.2. Note that $\mathbf{V}$ is intrinsic since it depends only on intrinsic features of $\Gamma(t)$. In the case of time-dependent region descriptors, other additive terms appear in (4.19) because of the two first terms in $\frac{dJ}{dt}$. But in the particular case of the Gaussian descriptors defined by (4.15) and (4.16), it is relatively easy to prove[4] that the two first terms in $\frac{dJ}{dt}$ are null, and that the speed that minimizes $J$ is still given by (4.19).

The segmentation process consists in applying the evolution equation defined by (4.18) and (4.19) to a user-defined initial condition:

$$(\Omega_{in}(0), \Omega_{out}(0), \Gamma(0)) = (\Omega^0_{in}, \Omega^0_{out}, \Gamma^0)$$

The result of the segmentation process is given by the "limit" of the triplet $(\Omega_{in}, \Omega_{out}, \Gamma)$.

## 4.5 Segmentation with *Fast-Marching* algorithm

If we consider in equation (4.5) the particular case of a motion equation with a speed function $F > 0$, hence the interface always moves outward. One way to characterize

[4] All the vector analysis theorems needed for the proof are recalled in [79].

the position of the interface is to compute the arrival time $T(\mathbf{x})$ of the interface as it crosses each point. Embedding this hypothesis in the level-set framework, a new equation is derived that determines the evolution of the surface $T(\mathbf{x})$ given by

$$\begin{aligned} T(C(\mathbf{x},t)) = t \quad &\Rightarrow \quad \nabla T \cdot C_t = 1 \\ &\Rightarrow \quad \nabla T \cdot \left( F \frac{\nabla T}{|\nabla T|} \right) = 1 \\ &\Rightarrow \quad F \cdot |\nabla T| = 1 \end{aligned} \tag{4.20}$$

that can be interpreted as: the gradient of arrival time is inversely proportional to the speed of the interface. This equation is the stationary case of the Hamilton-Jacobi formulation for propagating fronts where the time is disappeared, and if the speed is a function of the position of the front only, the equation reduces to what is known as the *Eikonal equation* equation, extensively studied in part I.

This particular equation (4.20) has also been used for surface extraction, since it has the same advantages as the *Level-Sets* formulation, detailed in section 4.3. In [111], *Malladi and Sethian* use the *Fast-Marching* algorithm in order to give a fast and rough initialization to a costly segmentation *Level-Sets* formulation. The *Level-Sets* model is based upon a composite flow, as shown in equation (4.11), for the segmentation of the cortex. The result of this segmentation was popularized by being advertised on the front cover of the *American Scientist Journal* [162]. They use as a speed function the following expression

$$F(\mathbf{x}) = e^{-\alpha |\nabla I_\sigma(\mathbf{x})|}, \; \alpha > 0 \tag{4.21}$$

where the speed has values very close to zero near high image gradients of the smoothed image $I_\sigma$. This expression is input in the *Eikonal equation* equation

$$|\nabla T(\mathbf{x})| = \frac{1}{F(\mathbf{x})} \tag{4.22}$$

The *Fast-Marching* in three dimension (see section 2.1) is then employed to march ahead. This helps to construct very quickly a good initial guess on the final surface. Then they input the final function $T(\mathbf{x})$ as an initial condition $\phi(\mathbf{x}, t = 0) = T(\mathbf{x})$, and iterate equation (4.11) a few time-steps with finite difference approximation schemes.

In practice the marching method is employed to march until a fixed time or until the size of the heap does not change very much (see section 1.3.2 for details on the heap used in the *Fast-Marching* algorithm) between two successive time increments.

## 4.6 Medical imaging applications of the *Level-Sets*

The *Level-Sets* models have already been used for a wide variety of applications, among them stereo problem [50], image classification [152], tracking of moving objects [78], or modeling deformations of solid objects [185] among others. But the most important set of applications of the *Level-Sets* has been done in medical imaging, where their interesting properties in handling topology changes are very useful for segmenting the very complex anatomical shapes.

### 4.6.1 Cortex segmentation

Since initial contributions on 3D segmentation of medical images using the *Level-Sets* models by *Malladi and Sethian* [110, 111], the very complex shape of the cortical surface has attracted numerous contributions, due to the interesting properties of the *Level-Sets* formulation. One example of brain tissue segmentation is shown in figure 4.10, where the algorithm used is the *Fast-Marching* on the basis of the work presented in [111].

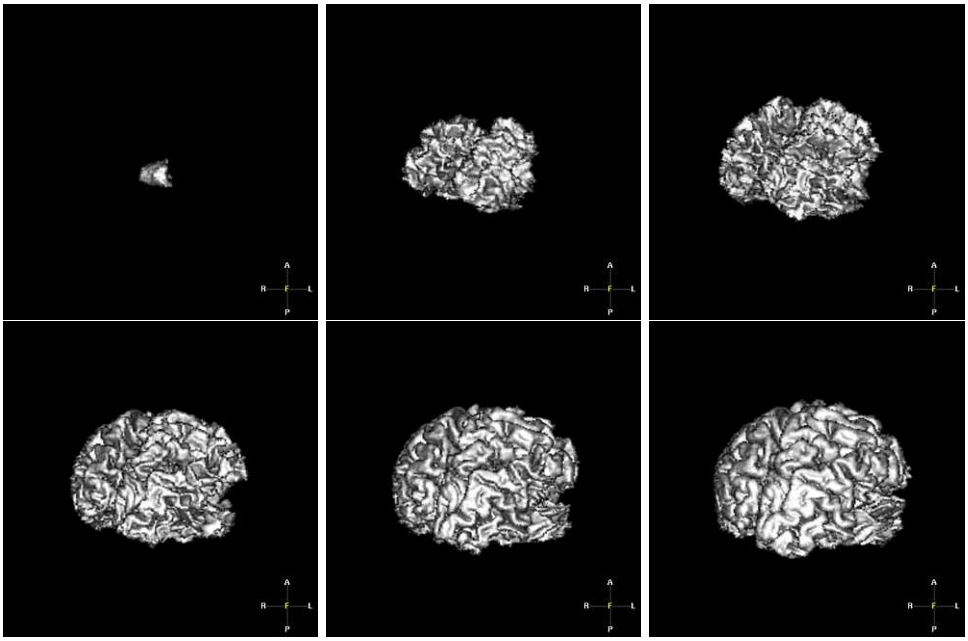


**Figure 4.10. Segmentation result on the brain:** On the basis of the model defined in [111], we segment the brain complex surface with the *Fast-Marching* algorithm, using a speed which is a simple distance function to an *a priori* mean value inside the white matter of the brain; formally $\mathcal{P}(\mathbf{x}) = \max\left(I(\mathbf{x}) - I_{mean}, 0\right) + w$.

In [67], authors proposed a fast implementation of the *Level-Sets* based on a semi-implicit formulation of the discretized evolution equation, based on similar use of these schemes for anisotropic diffusion filtering in [182].

An original method is presented in [7] where the authors propose to segment brain surfaces using sequentially a registration technique, and a *Level-Sets* model. However, it is not a real cooperation between both techniques, since the initialization of any parameter of the *Level-Sets* are tuned on the basis of the first results obtained through the registration process. We will see in the following that the *Level-Sets* can cooperate with other fast segmentation techniques to enhance their preliminary results.

Several works have been developed on the basis of the simultaneous segmentation of the grey and white matter of the brain, using *coupled curve evolution equations*

for segmenting both complex surfaces [193]. Main idea is to introduce a term relative to the distance between the two surfaces [194, 195]. In [68], this method is improved with a new scheme that computes the evolution of a new function which remains a distance function across iterations, and leads to better measurements of the distance between the two surfaces. And in [66] authors introduced the first geometric variational formulation for the coupled surfaces.

### 4.6.2 Brain Vessels

In the same region of the body, brain vessels extraction has attracted lots of attention from the computer vision community, with developments of very interesting dedicated *Level-Sets* implementations for thin tubular structure extraction.

We made a test on brain vessel extraction: the model is based on Gaussian region descriptors, and the initialization for $k_{in}$ and $k_{out}$ has been done using a **MDL** (minimum description length), with hypothesis that the image grey level information is a mixture of two Gaussian distributions. Descriptors are shown in figure 4.12. In figure 4.12 we show iterations of the segmentation of the brain vessels, in the

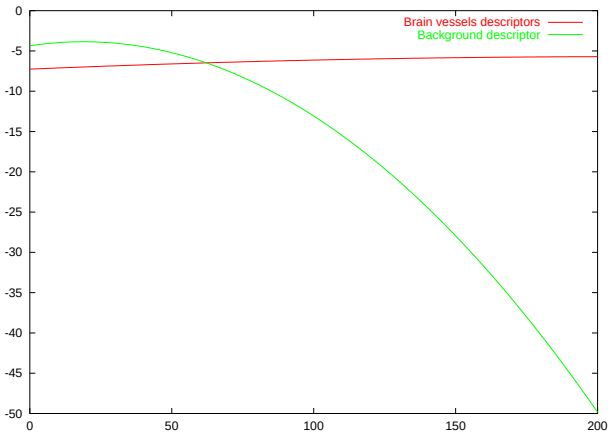



**Figure 4.11. Gaussian region descriptors for brain vessels**: assuming that the image is a mixture of two Gaussian distribution, the **MDL** classical computation of mean and variances has led to a model with ($\mu_{in} = 211.09, \sigma_{in} = 119.902$) and ($\mu_{out} = 18.9269, \sigma_{out} = 18.8822$).

dataset already displayed in figure 3.17, with the descriptors shown in figure 4.12. The propagation starts from one part of the vessels and propagates in the whole set. Computing time for extracting the complete brain vessels is more than an hour, an a standard Sun workstation (300 MHz cpu).

In this field, *Lorigo et al.* [107] have developed flows using the Hessian information [108], co-dimension 2 evolution scheme for the extraction of thin curves in 3D [106], with application to the brain vessels. Same target was followed in [177] where the authors use a flow based on the divergence of the gradient in the image. Both works achieve extraction of thin curves where even *Level-Sets* classical formulation, as shown

in figure 4.12, fails. However, the computing cost to achieve is too important. In the next chapter, we will settle another framework for segmentation in interactive computing time.

## 4.7 Summary

In this chapter we have explained an abstract deformable representation of the boundary between two regions, as used in [196]. The formulation of the evolution of the interface between the regions, defined with region descriptors, has been already studied for image segmentation in [137] with *Level-Sets* models. In the following we are going to develop improvements of this framework, introducing user interactivity on the interface, developments of algorithms to accelerate the speed of the computations, using the *Fast-Marching* first used for segmentation in [111]. On this basis we will show several applications of our framework.

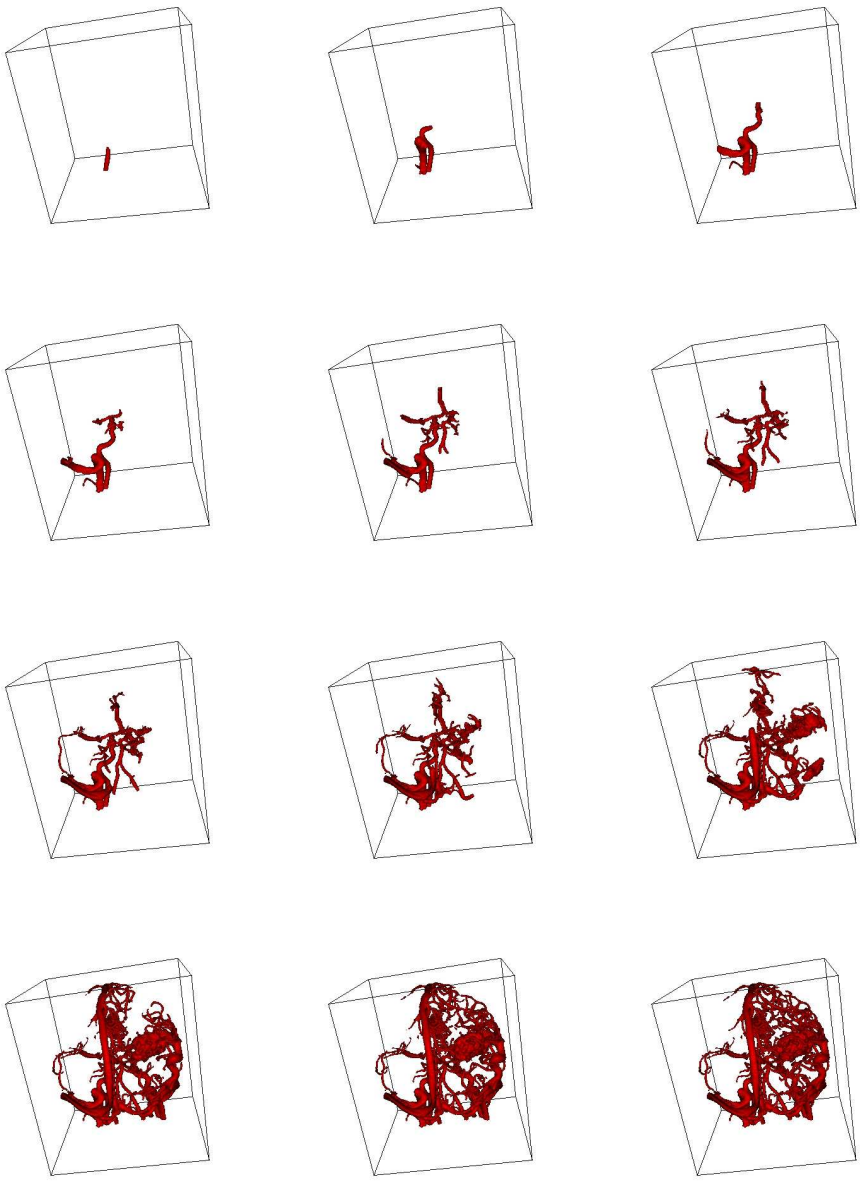


**Figure 4.12. Extracting the brain vessels with region-based forces**

# Chapter 5

# Several Segmentation Techniques involving *Level-Sets* and *Fast-Marching*

**Résumé** — En dépit de beaucoup d'avantages, les *Level-Sets* souffrent de sérieux inconvénients quand on les comparent à des représentations explicites ou paramétrées des surfaces actives. Dans ce chapitre, nous allons essayer de fournir une réponse à plusieurs de ces problèmes.
En ce qui concerne le manque d'interactivité, nous montrons comment implémenter quelques techniques d'interaction avec la surface implicite en section 5.1.
Par la suite nous présentons en section 5.2 une modification des forces de "région" sur lesquelles nous nous appuierons dans les applications.
Le coût de la segmentation d'une surface peut parfois être exhorbitant, lorsqu'on rajoute une dimension au problème comme c'est le cas avec les *Level-Sets* . Nous allons utiliser le *Fast-Marching* pour fournir une initialisation rapide et précise pour que le modèle plus complexe des *Level-Sets* n'ait besoin que de quelques itérations pour converger vers une solution encore plus précise. Dans les sections 5.3 et 5.4, nous présentons un algorithme de segmentation basé sur le *Fast-Marching* et la mise en oeuvre d'un algorithme combinant nos deux méthodes en une seule.

**Abstract** — Despite its advantages, level-set modelization suffers from several drawbacks compared with explicit and parametric models.
No local deformations are implemented, and in section 5.1, we introduce several techniques to include interactivity in the traditional *Level-Sets* framework. We also derive variations of the region-based forces, which will be much useful for our segmentation applications.
A large number of computations is often needed to solve the variational equations involved in the *Level-Sets* model. Using the *Fast-Marching* as a segmentation tool, we are going to initialize *Level-Sets* with a pre-segmentation near the solution, where only a few iterations are needed to converge to a sub-pixel accurate solution. In section 5.3 and 5.4, we present the collaboration of *Fast-Marching* and *Level-Sets* in a single segmentation framework.

## 5.1 Interactive Segmentation with the Level-Sets Framework

In medical imaging no automatic method is known to be generally applicable, completely reliable and robust. As a consequence, interaction is a part of many segmentation procedure to be combined to the automated segmentation process. No interactions were introduced in level-sets models, and the effect of the parameters on the model is often non-intuitive, as shown in figure 5.1.

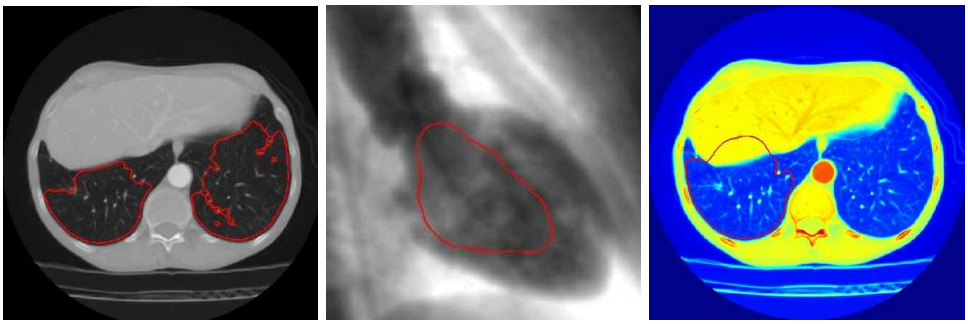


**Figure 5.1. Failed segmentation with Level-Sets paradigm** - Left image: The gradient influence is too important, and the level-sets is attracted by any spurious edges (vessels and arteries in the lungs); middle image: the curvature forces being too important, the inflation force is unable to reach the edges of the left ventricle; on the lung image the edges are now not sufficiently valued in order to stop inflation into the liver which surrounds the lung.

### 5.1.1 Interactivity requirements for the level-sets paradigm

As emphasized by most of the comparative study of explicit and implicit deformable models, and particularly in the PhD thesis of *J. Montagnat* [120], the formalism of the *Level-Sets* implicit representation restricts severely user interactivity.

The interactive requirements, when an automatic method fails, can be classified in different categories, like

- unseen evidence: the user perceive an edge where the automatic method does not (when the object intensity does not fit). In this case, the method should offer a regionally different edge defining operation or a regional parameter tuning (all examples in figure 5.1 are cases of unseen evidence);
- absence of evidence: neither the user nor the automatic method observe an edge. Interaction should imply a nullification of the evidence from the data in the region;
- dislocated evidence: the automatic method has found proof of an object according to the object model, but has confused the object with a neighboring object. Interaction should confine the admissible contours to the indicated zone.

The definition of the correct interactivity framework is more than a difficult task, maybe more than the settle of an automatic method for medical imaging problems. The tremendous work of *S. Olabarriaga* [133,134] on the subject settled the basis for a general interactive segmentation framework, but solutions surely lie in dedicated approaches to specific problems.

Simple interactions have been implemented in 2D for tests. They are based on modifications of the data or the model, which are two cases to answer the problems of unseen evidence and absence of evidence.

### 5.1.2 Fixing a point of the contour

This force will attract the zero level-set of $\phi$ to a user defined point. If we apply the classical evolution equation (4.5) of [135], with an added external force to the point $\mathbf{x_0}$:

$$\phi_t \quad + \quad \tilde{F}|\nabla\phi| = \phi_t + F|\nabla\phi| + \mathcal{F}_{\mathbf{x_0}}|\nabla\phi| = 0 \tag{5.1}$$

with the external force compute at each pixel $\mathbf{x}$ by

$$\mathcal{F}_{\mathbf{x_0}}(\mathbf{x}) = d(\mathbf{x_0};\mathbf{x}) \tag{5.2}$$

This force is applied to the nearest voxels of $\mathbf{x_0}$ (in a user-chosen neighborhood), on the zero level-set. In order to compute $\mathcal{F}$ for an undefined number of fixed point $\{\mathbf{x_0},\dots\mathbf{x_N}\}$ in equation (5.2), we use the *Fast-Marching* algorithm, detailed in chapter 1, as explained in table 5.1. Figure 5.2 represents iterations of the evolution

**Algorithm for Computing Interactivity Force $\mathcal{F}$**
at iteration $i$, let $\{\mathbf{x_0},\dots,\mathbf{x_N}\}$ be the $N$ user defined fixed points, and $N$ action maps $U_0,\dots,U_N$. For $j \in [1;N]$:

- $U_j(\mathbf{x_j}) = 0$, and $U_j(\mathbf{x}) = \infty$ elsewhere;
- propagate a front by computing action map $U_j$ with $\|\nabla U_j\| = 1$ using *Eikonal equation* (1.6);
- stop when visiting a pixel $\mathbf{y}$ where $\phi$ sign changes.

The additional force $\mathcal{F}$ is given by $\mathcal{F}(\mathbf{x}) = \min_{j\in[1;N]} U_j(\mathbf{x})$ if $\mathbf{x}$ has been visited, in other case $\mathcal{F}(\mathbf{x}) = 0$.

**Table 5.1.** *Fast-Marching* for Computing external forces

process with *On-The-Fly* user interaction of a *Level-Sets* initialized by the distance to a circle, and using as evolution equation:

$$\phi_t + \mathcal{F}_{\mathbf{x_0}}|\nabla\phi| = 0 \tag{5.3}$$

where $F = \mathcal{F}$ in equation (5.1). Several comments can be added to this algorithm:

- choosing $\mathcal{F}(\mathbf{x}) = \min_{j\in[1;N]} U_j(\mathbf{x})$ instead of $\max_{j\in[1;N]}(U_j(\mathbf{x})$ in table 5.1, is just a matter of convention. The force applied is not directional, because the

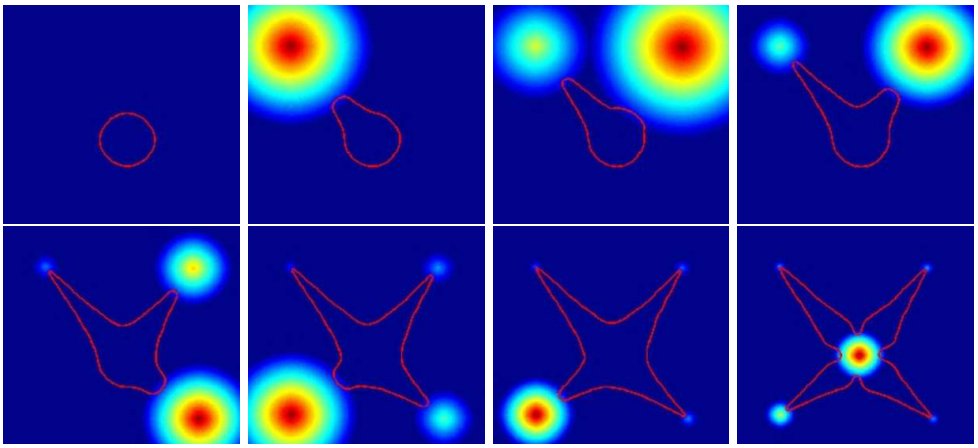


**Figure 5.2. Fixing a point of the contour**: these images represent the deformation of the zero level-set of $\phi$, when fixing different points of the contour. The zero level-set extracted is represented in red, and is super-imposed to the external force $\mathcal{F}$ in equation 5.2

evolution of the curve is in its normal direction and do not integrate any tangential term. Therefore, the zero level-set is not attracted more or less by two neighboring pixels, and this is just the intensity of this force that can be tuned;

- concerning the different action maps $U_0, \ldots, U_N$, they can be integrated in only one action map $U$, according to the previously mentioned convention. This can greatly reduce the computing cost of $N$ float images, for each action map. This is easily handled by taking the minimum of the different actions, because the *Fast-Marching* algorithm has been built on this minimality principle. It was the method used in figure 5.2.

### 5.1.3 Re-initializing the contour

The interaction we study now is a re-initialization of the contour, on the model of the *Convolution surface* developed in [14], for implicit function modeling in computer graphics (see [13]). A convolution is a modification of a signal by a filter; here the skeleton is the signal and the filter is a Gaussian kernel. More than a deformation process, this is a re-initialization of the model.

Considering our function $\phi$ defined on a small narrow-band, as done by *Whitaker* [184], with its *Sparse-Fields* method the level-set function is close to a binary image, as shown in figure 5.3. Defining a point $\mathbf{x_0}$ for the interaction, we compute its distance to the zero level-set, and its corresponding nearest point $\mathbf{x_1}$ on this zero level-set. In our 2D example, the segment $[\mathbf{x_0}, \mathbf{x_1}]$ is the skeleton that we convolved with a Gaussian kernel. We built a function $\mathcal{F}$ that for any point $\mathbf{p}$ on the image domain

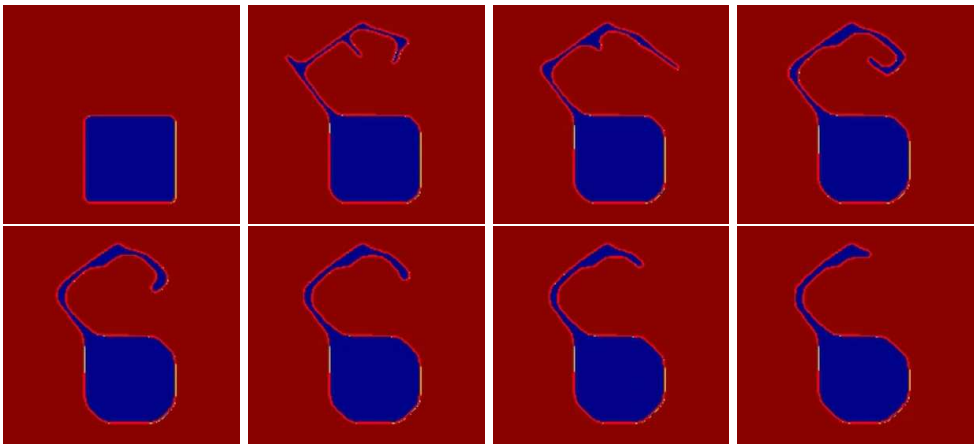


**Figure 5.3. Re-initializing the contour**: The initial contour which evolves according to an equation of the form $\phi_t + \kappa|\nabla\phi| = 0$ is deformed by convolution. Different steps show the curvature motion that shrinks the obtained closed curve, as explained in [70].

gives

$$\mathcal{F}(\mathbf{p}) = a\left(\int_{x_0}^{x_1} e^{-\|\mathbf{p}-\mathbf{u}\|^2/2b} du - 1/2\right) \tag{5.4}$$

that can be approximated by

$$\mathcal{F}(\mathbf{p}) = a\left(\sum_i e^{-\|\mathbf{p}-\mathbf{x_i}\|^2/2b} - 1/2\right) \tag{5.5}$$

where $\mathbf{x_i}$ are the points on the skeleton, and $a, b$ are positive constants to control the amplitude and sharpness of the convolved object. Then we apply $\mathcal{F}$ by reconstructing the new function $\varphi$ defined by the two regions:

- $\varphi^- = \phi^{-1}(\mathbb{R}^-) \cup \mathcal{F}^{-1}(\mathbb{R}^-)$
- $\varphi^+ = \phi^{-1}(\mathbb{R}^+) \cap \mathcal{F}^{-1}(\mathbb{R}^+)$

where

- if $\mathbf{x} \in \varphi^-$, $\varphi(\mathbf{x}) = \max(\phi(\mathbf{x}), \mathcal{F}(\mathbf{x}))$;
- if $\mathbf{x} \in \varphi^+$, $\varphi(\mathbf{x}) = \min(\phi(\mathbf{x}), \mathcal{F}(\mathbf{x}))$.

Applying this method to $\phi$, we modify the zero level-set as done in figure 5.3. This implementation of a re-initialization is rather similar to the method proposed in [141].

### 5.1.4 Conclusion on the interactivity

We have shown two possible interactivity techniques, based on a modification of the data, and on a modification of the model, both based on computer graphics ideas. Basically, a lot of techniques first studied for computer graphics can be applied to the *Level-Sets* which basis is an implicit representation. The implicit representation in computer graphics (see [13]) has much in common with the formalism developed in [135]. It was even studied by *Cani and Desbrun* in [19] and detailed in [41], with a *Level-Sets* implementation for a "skin" that contains spherical particles for the simulation of highly deformable models. But in Computer Graphics, first matter is the rendering of animations, and not their computing costs. Unfortunately, interactivity requires a direct and fast output of any user interaction, which is something that is still difficult to manage with the *Level-Sets* paradigm.

But the solution probably lies in the combination of two different techniques: one for segmentation, and one for interaction. A very good example of this philosophy can be found in [188]: the authors use circular oriented particles to sample and control implicit surfaces. This technique is also used by *Szeliski et al.* [171]. The model is a force-feedback system where particles try to match the zero level of the implicit function, and where the surface follows particles. A simple constraint locks the set of particles onto a surface while the particles and the surface move. The particles use mutual repulsive forces and fissioning to sample the whole surface, and the user interactivity can be applied locally on one particle at a time, using them as control points for direct manipulation, as shown in figure 5.4[1]. Finally, very interesting work

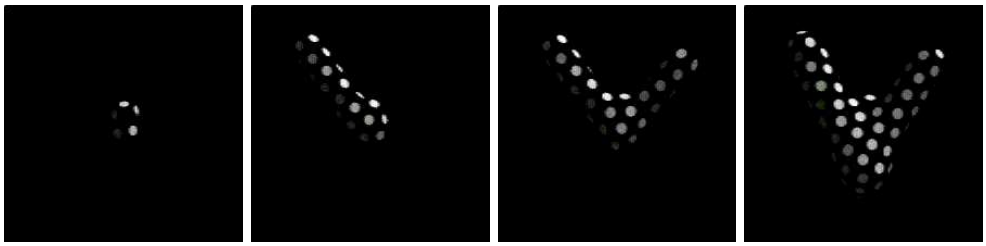


**Figure 5.4. Interactivity on an implicit surface** : the particles track the implicit surface, leading to an easy visualization of its zero level-set, and at the same time input a control on this surface, in order to modify its shape.

has been done on the construction of subjective contours in [154], where the *Level-Sets* extracts contours in famous test images of *Kanizsa* [81] that strongly requires image completion.

[1] We would like to acknowledge Andrew Witkin and Paul Heckbert, for providing this sequence that is an illustration of their article [188].

## 5.2 Region Based Forces

### 5.2.1 Gaussian descriptors

We recall the descriptors first introduced by *Zhu and Yuille* [196]. They are "fully automatic" in the sense that no user-defined parameter is necessary to their definition. They are based on region probabilities defined by Gaussian distributions:

$$P_{in}(x,t) = \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(I(x)-\mu_{in}(t))^2}{2\,\sigma_{in}^2(t)})$$

A similar expression is used for the definition of $P_{out}(x,t)$. Here $\mu_{in}(t)$ and $\sigma_{in}^2(t)$ are respectively the mean and the variance of the image intensity over $\Omega_{in}(t)$:

$$\mu_{in}(t) = \frac{\int_{\Omega_{in}(t)} I(x)\,dx}{\int_{\Omega_{in}(t)}\,dx}\ ,\ \sigma_{in}^2(t) = \frac{\int_{\Omega_{in}(t)}(I(x)-\mu_{in}(t))^2\,dx}{\int_{\Omega_{in}(t)}\,dx}$$

This means that the histograms of the intensity in $\Omega_{in}(t)$ and $\Omega_{out}(t)$ are *modeled* by Gaussian distributions.

### 5.2.2 Modified Gaussian descriptors

The preceding model has one major drawback: if one of the variances is much smaller than the other one, then undesired results can be observed. For example, if the image is composed of a bright region with a small variance (a contrast-enhanced organ for instance) in the middle of a dark background with a large variance, then very bright pixels may have a greater probability to be in the background than in the bright region. This is perfectly normal from the point of view of the Gaussian model, but it is in contradiction with our *a priori* knowledge about the image informational content. In other words, the Gaussian model may be unadapted to some images.

In such cases, we can introduce an priori knowledge by modifying the two Gaussian distributions. If we know that the image region that we want $\Omega_{in}(t)$ to cover is supposed to be brighter than the one covered by $\Omega_{out}(t)$, then we take:

$$P_{in}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(I(x)-\mu_{in}(t))^2}{2\,\sigma_{in}^2(t)}) & \text{if}\quad I(x) < \mu_{in}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)} & \text{if}\quad I(x) \geqslant \mu_{in}(t) \end{cases}$$

and:

$$P_{out}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)} & \text{if}\quad I(x) < \mu_{out}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)}\ \exp(-\frac{(I(x)-\mu_{out}(t))^2}{2\,\sigma_{out}^2(t)}) & \text{if}\quad \mathcal{I}(x) \geqslant \mu_{out}(t) \end{cases}$$

It is also possible to manually choose the means and variances instead of computing them from the region histograms, which can be very useful in some applications (they can be obtained through an initialization process). In this case, the region descriptors become time-independent.

### 5.2.3 Sigmoid descriptors

For the images where the modified Gaussian model failed, we used user-defined time-independent sigmoid probabilities. We took:

$$P_{in}(x,t) = \frac{1}{1+e^{a_{in}\,(b_{in}-I(x))}} \tag{5.6}$$

with a similar expression for $P_{out}(x,t)$. While Gaussian have quadratic logarithms, sigmoid functions have asymptotic linear logarithms. For example, if $a_{in}$ is positive, then:

$$\lim_{I(x)\to+\infty} \log P_{in}(x,t) = 0$$

and:

$$\log P_{in}(x,t) \sim_{I(x)\to-\infty} a_{in}\, I(x).$$

Those descriptors for the aorta of figure 3.12 are shown in figure 5.5.

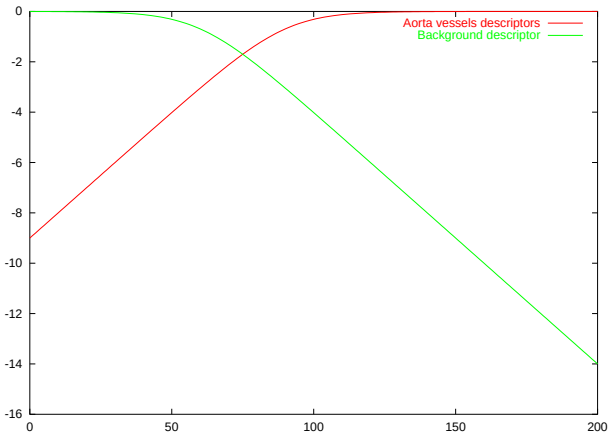



**Figure 5.5. Sigmoidal region-based forces**: they are used for discriminating the aorta towards the background in figure 3.12.

Figure 5.6 shows iterations of the segmentation of the aorta of figure 3.12 with sigmoidal region descriptors of figure 5.5.

### 5.2.4 Numerical implementation of the level-sets paradigm

#### Adaptive time step

We used an Euler first-order explicit time scheme for solving (4.6). We recall that explicit time schemes often require a limitation of the time step to be stable. Here it is possible to control the Courant-Friedrichs-Lewy (or CFL) number[2] of the discretized flow equation, by adapting the time step between each time iteration.

[2]Number of cells crossed by a characteristic curve during a time step.

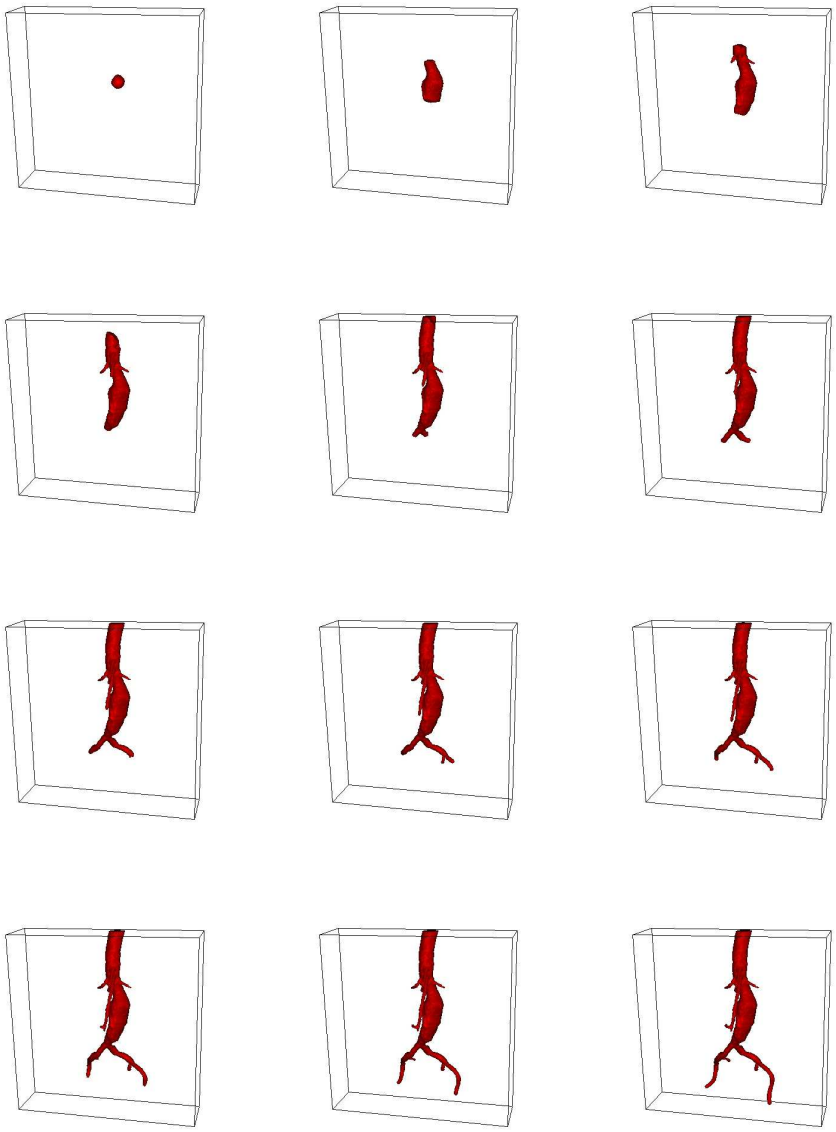


**Figure 5.6. Extracting the aorta with region-based forces**: the initialization is a sphere located inside the aorta object. Using the *a priori* distribution of the grey levels inside the aorta and boundary based forces, we segment the aorta, as shown on the several iterations.

The only difficulty is related to the scheme associated with the curvature flow, which has no obvious stability conditions. We have empirically chosen to give a stronger penalty to its contribution to the global CFL number. Explicitly in 3D, for a composite flow of the following form:

$$\mathbf{V} = (\beta(\mathbf{x},t) - \varepsilon(\mathbf{x},t)\,\kappa_M(\mathbf{x},t))\mathbf{n} + \mathbf{U}(\mathbf{x},t)$$

we ensure that:

$$\max_{(i,j,k)} (\nu^{\beta}_{ijk} + 3\,\nu^{\varepsilon}_{ijk} + \nu^{U}_{ijk}) \leqslant 1$$

where:

$$\begin{aligned}
\nu^{\beta}_{ijk} &= \Delta t\,|\beta_{ijk}|\,\sqrt{\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2}} \\
\nu^{\varepsilon}_{ijk} &= \Delta t\,|\varepsilon_{ijk}|\,\sqrt{\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2}} \\
\nu^{U}_{ijk} &= \Delta t\,\left(\frac{|U_{x\,ijk}|}{\Delta x} + \frac{|U_{y\,ijk}|}{\Delta y} + \frac{|U_{z\,ijk}|}{\Delta z}\right).
\end{aligned}$$

The results are very satisfying, and no numerical instability has been noticed during the segmentation tests.

### Narrow Banding

For reasons of causality, it is possible to restrain the computation domain to a band of cells around the zero-level set of $\phi(\cdot,t)$. The result is a decrease of the computational cost. Several approaches have already been proposed, but we have build a new one which is adapted to our segmentation problems.

Classical approaches are referred to as narrow-band methods (see figure 5.7). Some are based on combinatory studies in order to add or remove cells around $\phi(0,t)$ as it moves (see *Adalsteinsson and Sethian* [2]). Another approach can be found in the thesis of *Keriven* [84], and consists in computing the signed distance function to $\phi(0,t)$ when it gets too close to the boundaries of the band, and re-initializing $\phi(\cdot,t)$ with the computed distance function. Other authors chose to initialize $\phi$ with a distance function and maintain $\|\nabla\phi\| = 1$ at each iteration (see *Adalsteinsson and Sethian* [3], and *Gomes and Faugeras* [68]).

These methods cited above are efficient, especially when the initial condition $\phi(\cdot,0)$ is far from the solution. Here we propose a new specific method that is very efficient for the segmentation of fine- to medium-size tubular structures (like arterial, venous, or bronchial trees), or when using a rough pre-segmentation for initializing $\phi$.

The idea is simple: we start from a distance function, either associated to a pre-defined geometric primitive, or computed from a binary pre-segmentation by a fast-marching algorithm (see next chapter). We define a narrow-band area and a wide-band area around $\phi(0,0) = \phi_0$ (the wide-band is larger than narrow-band and comprises it). The calculations are then performed in every cell of the wide-band area. If $\phi(0,t)$ gets out of the narrow-band area (i.e. the sign of $\phi$ changes somewhere

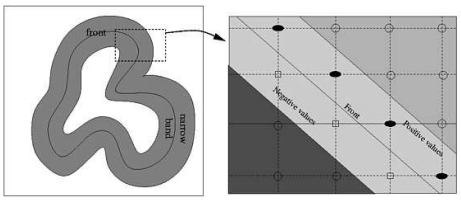



**Figure 5.7. Narrow Band Diagram:** Left image shows the overview of the narrow-band structure, and right image shows the detail.

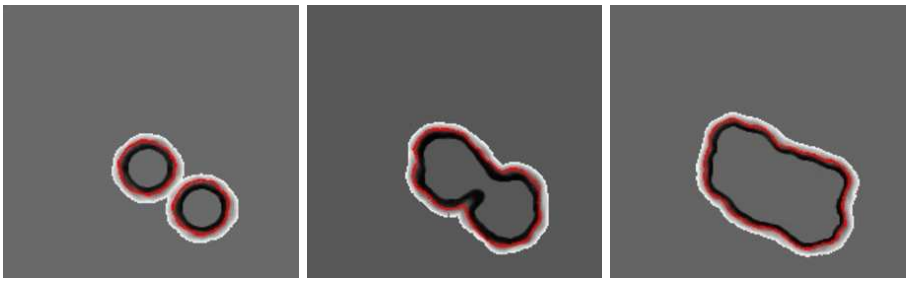


**Figure 5.8. Narrow Band reconstruction:** images are samples of the evolution and reconstruction of the narrow-band, starting from two circles that merge.

in the outside of the narrow-band area), then both bands are locally enlarged in a spherical manner.

As a consequence, we can add cells to the computational domain, but we never remove cells from it. If a pre-segmentation is used, the enlargements of the bands are few and the resulting computational cost is very small. This method has also proved to be efficient in the case of propagation in tubular structures with diameters comparable to the wide-band width (for example 10 times a cell diagonal).

#### Boundary conditions

We used so-called *free* boundary conditions, that is to say that we extrapolate the values of $\phi(\cdot, t)$ in the outside vicinity of the computation domain boundaries by taking the closest value inside the domain, and that the corresponding finite differences are equal to zero. Some authors use other entropic boundary conditions. The main idea is that the information that goes out of the computation domain is lost, and that no information should be introduced into the domain through its boundaries.

## 5.3 Using the Fast-Marching for segmentation

Originally introduced by *Malladi and Sethian* [111], the *Fast-Marching* can be used as a fast initialization algorithm for image segmentation. Despite the fact that it requires low computational cost, it has several drawbacks, in particular that the speed $F$ in

equation (4.20) is always positive or always negative. Therefore, the front evolves and propagates in the whole image domain, without being stop by any edge, and it cannot be used for cases with curvature-dependent speed functions. The stopping criterion still relies on a user perception of the segmentation, like visual inspection of the domain visited by the algorithm during propagation, or the choice for an appropriate stopping time, as done in [111]. But the nice segmentation properties of the *Fast-Marching* algorithm shall be taken into account, as shown in figure 4.10, where we segment a brain on the basis of the method described in [111].

Still, the stopping criterion in this case was to visually control the segmentation process, because the automatic detection of the crossing of the interesting edges is difficult to implement.

Noticing that *Eikonal equation* is used to model the continuous formulation of the watershed transform, we have developed an interactive segmentation tool based on the flooding process of this morphological algorithm introduced in [180]. It is based on the same principle that segmentation boundaries are defined by pixels where several evolving fronts collide.

### 5.3.1 Comparison with the watershed algorithm

The classical morphological segmentation paradigm [156] is based on the watershed transform [181]. The method is very simple:

- the gradient image $\|\nabla I\|$ is computed (and enhanced);
- for each object of interest, an inside particle is detected (either interactively or automatically);
- flooding waves are propagated from the set of markers and flood the topographic surface $\|\nabla I\|$.

The points where the flooding waves meet each other form the segmentation boundaries. Different regions between boundaries are called catchment basins. Usually markers are the regional minima of the gradient image. An example of the watershed transform is shown in figure 5.9. Very often, the minima are extremely numerous, leading to an over-segmentation, therefore for practical cases, the watershed transform will take as seed points a smaller set of markers, identified through pre-processing techniques.

The flooding waves are propagated according to the topographic map $\|\nabla I\|$, where the pixels belonging to a catchment basin are nearer to its corresponding regional minimum than to any other minima. In this framework, the construction of catchment basins can be seen as a shortest path problem between a marker and the image points. It corresponds to compute the gray-weighted distance transform (GWDT) of the image to the set of markers. There are several ways to compute this $GWDT$ : as shown in [118], viewing the topographic image domain as a refractive index, as explained in chapter 1, the distance between a marker and any point in the image is the line integral of the penalty $\|\nabla I\|$ along the optical path length. We reformulate *Eikonal equation* 4.22 with

$$\|\nabla T(\mathbf{x})\| = \|\nabla I(\mathbf{x})\| \tag{5.7}$$

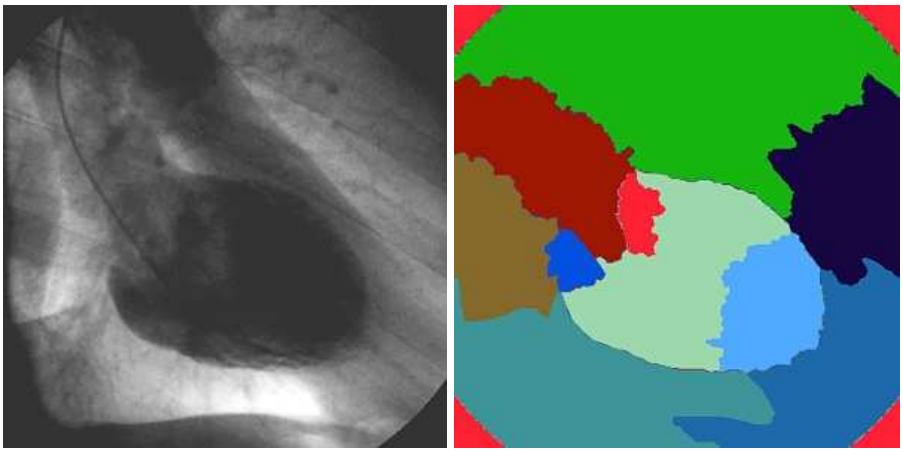


**Figure 5.9. Watershed transform of a 2D X Ray image of the heart (LV)**: The seed points are regional minima of the gradient of image 5.9-left; the flooding was implemented in the continuous framework of *Eikonal equation* , and the regional minima were filtered in order to have a smaller set of markers.

and the *GWDT* can be computed on the whole image domain, using the markers as sources pixels $p$ where $T(\mathbf{p}) = 0$.

*Maragos* has shown in [118] that the continuous segmentation approach based on the *Eikonal equation* outperforms the discrete segmentation results. However several problems remain important:

- the final segmentation relies on the number of the markers: too many markers lead to an over segmentation of the image, and merging algorithms are needed as post-processing;
- it relies also on the quality of the edge detector applied to the real image: if there is a gap in the edge information, the nearest marker will flood the adjacent region.

We have devised a very simple method based on user interactivity, and different front speeds, based on the formulation of the flooding of the whole image. Each front, initialized by a user-defined seed point, will propagate according to its own propagation speed, derived from local image information (and not just gradients), and will stop when it meet other propagating fronts. This method has been implemented to provide a *quick and dirty* initialization for the *Level-Sets* method.

### 5.3.2 Interactive segmentation with the Fast-Marching

#### Flooding algorithm

In the following, we assume that we have an undefined number of seed points for front propagation, that can be labeled with a known number of labels. We detail below the

flooding algorithm for multi-label front propagation.

**Definition**

- a set of starting point $\mathbf{p}_1, \ldots, \mathbf{p}_n$, located inside the image domain;
- each starting point $\mathbf{p}_i$ is assigned a label $l_i$ $i \in [1; n]$ (not necessarily different);
- a label map $\mathcal{L}$, for controlling collision;
- a penalty image $\mathcal{P}$ which will drive the front propagation, and which is a function of the position and the label of the front;
- We initialize the usual set of data-structures for front propagation, including an action map $\mathcal{A}$ and (only) one min-heap structure $\mathcal{H}$;

**Initialization**

- we initialize a classical front propagation method, setting $\mathcal{A}(p_i) = 0$ $\forall i \in [1; n]$ and storing all seed points in the min-heap structure;
- $\mathcal{L}(\mathbf{p}_i) = l_i$, $\mathcal{L} = -1$ elsewhere;

**Loop:** at any iteration

- Let $\mathbf{x}_{min}$ be the *Trial* point with the smallest action $\mathcal{A}$;
- Move it from the *Trial* to the *Alive* set (i.e. $\mathcal{A}_{\mathbf{x}_{min}}$ is frozen);
- Update $\mathcal{P}(\mathcal{L}(\mathbf{x}_{min}))$
- For each neighbor $\mathbf{x}$ (6-connexity in 3D) of $\mathbf{x}_{min}$:
  - If $\mathbf{x}$ is *Far*
    1. add it to the *Trial* set;
    2. $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{x}_{min})$
    3. compute $\mathcal{A}$ according to equation

$$\|\nabla \mathcal{A}\| = \mathcal{P}\left(\mathcal{L}(\mathbf{x}_{min})\right) \tag{5.8}$$

       using the up-wind discretization scheme, involving only the neighbors of $\mathbf{x}$ with label $\mathcal{L}(\mathbf{x}_{min})$;
  - If ($\mathbf{x}$) is *Trial*
    1. recompute the action $\mathcal{A}(\mathbf{x})$ with equation( 5.8), involving only the neighbors of $\mathbf{x}$ with label $\mathcal{L}(\mathbf{x}_{min})$;
    2. if the new value is smaller, then $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{x}_{min})$

**Termination**

- stop when all pixels on the image domain are visited;
- the label map $\mathcal{L}$ represents the final segmentation into $k$ regions $R_k$.

In this algorithm, we have deliberately left blank the update procedure of the potential. Using a very simple potential based on the grey level information. Having the $n$ seed points $\mathbf{p}_1, \ldots, \mathbf{p}_n$, we can set at initialization

$$\text{if } \mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{p}_i) \text{ , } \mathcal{P}(\mathbf{x}) = \max(I(\mathbf{x}) - I(\mathbf{p}_i), 0) + w \tag{5.9}$$

Then the speed is inversely proportional to the difference between the grey-level of the starting seed point and other points in the image. Result of this strategy can

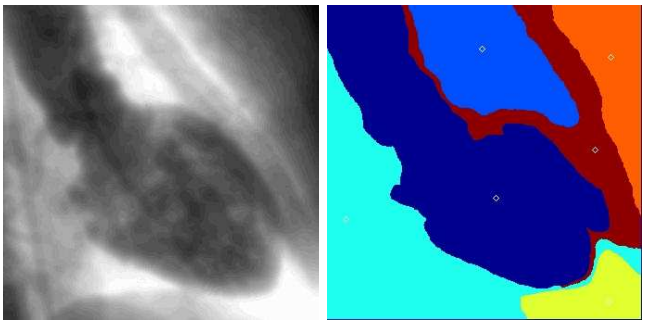


**Figure 5.10. Front competition on a 2D X Ray image of the heart (LV)**: The seed points were manually located on the dataset on the left image; they are visible as white dots on the right image.

be observed in figure 5.10 where different seed points were manually located on the image 5.10-left, leading to the segmentation of figure 5.10-right. The resulting is not convincing, but the use of this kind of information to differentiate the different front speeds has a bigger potential than the classical watersheds formulation. If there is a gap in the gradients between two basins, one of the two fronts could flood completely into the adjacent basin with the watersheds, whereas the speed in the adjacent region could still be discriminating the unwanted flooding, in the case of our competitive fronts algorithm. This is mainly the difference between a difference of a *plateau* and a barrier in the penalty (where *plateau* is a region with small variations in the intensity).

Thus, different strategies can be implemented, as many as there are different possible potentials.

### Setting the penalty information

We can devise a very simple algorithm that do not use the information coming from the seed points, but for the $m$ first visited voxels for each label. This reduces greatly the influence of the user interaction in setting the markers position. Therefore for each label,

- At iteration $i$ in the *Fast-Marching* we examine the point $\mathbf{x}$ in the min-heap whose action is minimal;
- We remove it from the heap and we examine its label $l = \mathcal{L}(\mathbf{x})$ and its grey level $I(\mathbf{x})$;
- For the front points with label $l$, we know $N_l$ the number of points considered and $\mu_l$ the mean grey level value of those points;
- if $N_l < m$, we update this mean grey level value for the label $l$

$$\mu_l = (N_l * \mu_l + I(\mathbf{x}))/(N_l + 1) \ , \ N_l + + \tag{5.10}$$

- considering that the image grey level of the pixel in the desired region are a random distribution of mean $\mu_l$ we can also update the variance with *Koenig-Huyghens* formula $\sigma_l^2[X] = \mu_l[X^2] - \mu_l[X]^2$ for the random variable $X$;

- we use this new mean values $\mu_l$ and $\sigma_l$ in the computations of the action at each neighbors of $\mathbf{x}$.

In figure 5.11 is shown an example of using this strategy for setting the penalty term. The speed function in the image shown is computed with $m = 10$ for all

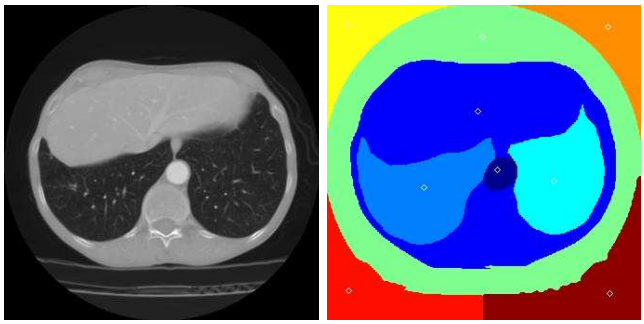


**Figure 5.11. Front competition on a 2D scanner slice of the lungs**: The seed points were manually located on the dataset on the left image; they are visible as white dots on the right image.

labels. The speed function for each label $l$ is computed with the penalty $\mathcal{P}(\mathbf{x}) = e^{-|I(\mathbf{x})-\mu_l|/2\sigma_l^2}$.

Unfortunately, this algorithmic trick makes the penalty a time-dependent function, and thus the minimality principle is violated, and *Eikonal equation* discretization cannot be applied in theory during the iterations where $\mathcal{P} = \mathcal{P}(\mathbf{x}, t)$. Therefore, this heuristic must be seen as a region growing algorithm to operate a statistical study at the beginning of the segmentation process before visiting the whole image domain. It can also be replaced by a similar statistical study in a sphere around the seed points.

Figure 5.12 displays the result of the same strategy on the brain dataset used for test in figure 4.10. This result was obtained by setting different seed points in the different objects in the brain MR image which is represented on the left column of figure 5.12. Each one of the regions was initialized with one marker, and they are represented when all fronts collide super-imposed on three orthogonal views of the dataset in the left column of figure 5.12. Inconvenient of this kind of segmentation is that it assumed that the correct number of classes is known *a priori* and that the user can clearly indicate to the algorithm the location of each connected components of this class. But target here is to provide a quick and dirty initialization to a more complex and time consuming method (like *Level-Sets* ).

The same test was done on a lung image, where we try to separate the airways from the soft tissue and vessels, in figure 5.13. Results The underlying dataset is a volume-of-interest extracted from a multi-slice CT scanner of the lungs. The dataset, almost isotropic, contains a huge number of pixels, and even a fast algorithm like ours is not able to achieve this result in “interactive time”. But we have shown the ability of this simple algorithm to achieve difficult segmentation tasks.

Further, in this algorithm, there is no need to use several data-structures (like min-heap) to store each different front. Due to the minimality principle of the *Fast-*

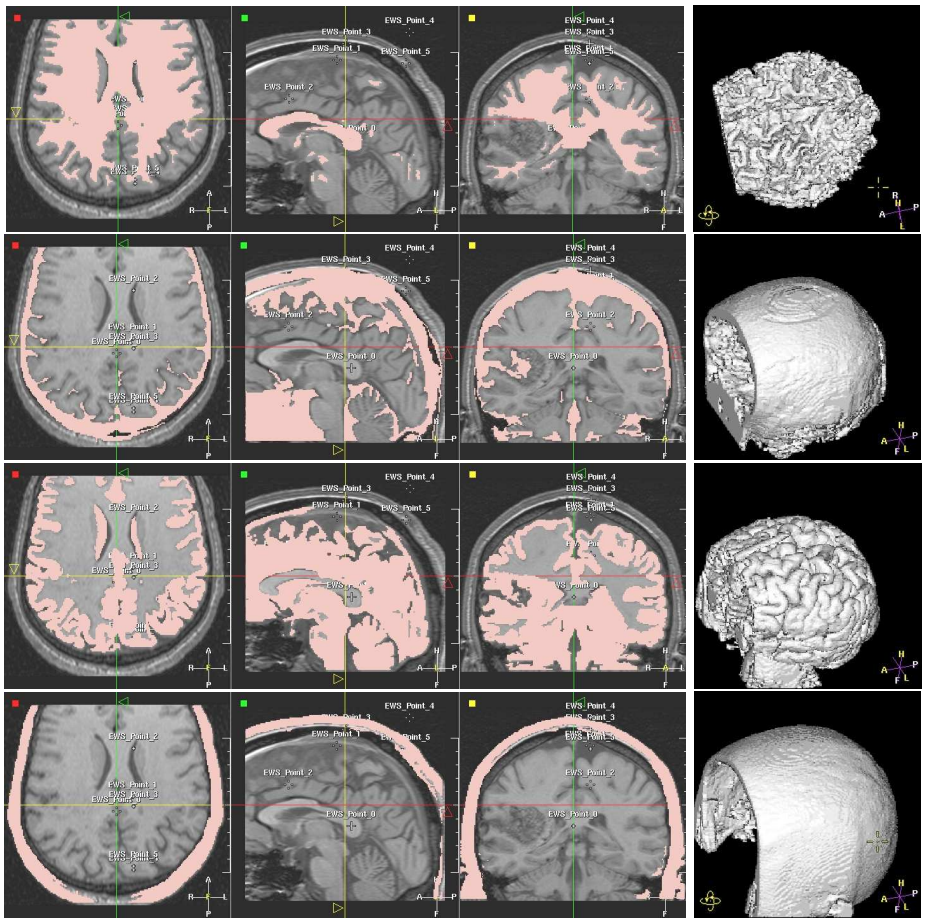


**Figure 5.12. Front competition result on the brain:** Simultaneous segmentation of the white and grey matter of the brain, plus the skull and the skin; each row corresponds to a region, and the intersection between the regions and the datasets is represented in pink, in the left column.

*Marching* construction, the point with minimum energy, independently from its label, will still be at the root of the min-heap data-structure. And each point of the image will be visited only one time, leading to approximately the same computing cost than the propagation of a single front in the image domain. It seems that a similar version of the multiple front propagation algorithm has been developed in [103, 167], for initializing color and texture segmentation with the *Level-Sets* methods.

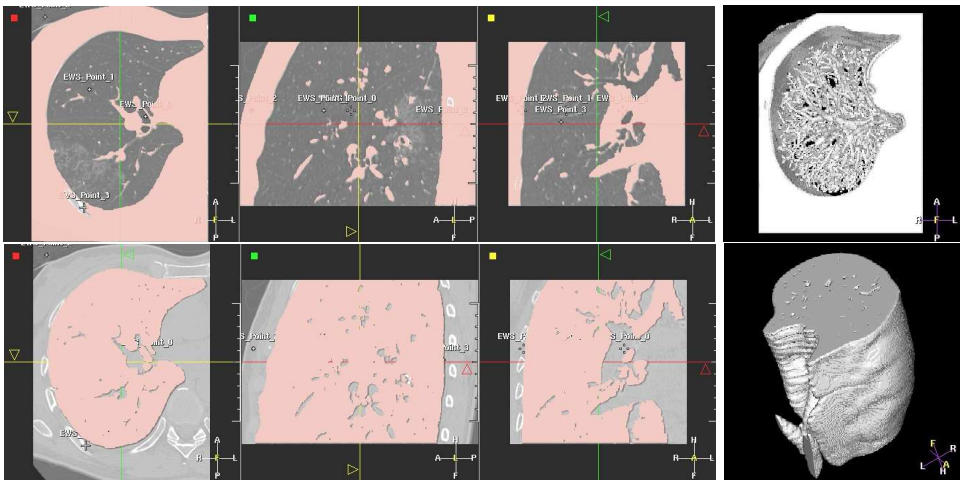


**Figure 5.13. Front competition result on the lungs**: separating air and soft tissues in a multi-slice CT dataset of the Lungs; seed points are manually located in the different parts of the image (on in the airways, on in the tissues, and on outside the object).

## 5.4 Combining Fast-Marching and Level Sets

### 5.4.1 Advantages and Drawbacks of *Fast-Marching*

The *Fast-Marching* algorithm is used for the computation of a minimal action map, or weighted distance transform (from the morphological point of view).

Drawbacks : monotonicity of the speed (can only propagate in one direction), no control of the curvature of the contour to be extracted, no stopping criterion (speed always strictly positive/negative).

#### Advantages

- It is a very fast algorithm (as shown in the case of path extraction for virtual endoscopy in chapter 3);
- it can segment objects with complex topologies (see the brain example in figure 4.10);
- it can be used with simple initialization of seed points (see the competitive front segmentation of figure 5.12);
- There are lots of Penalty definition (depending on user's imagination, ranging from grey-levels to Hessian measures [60], and other complicated measured obtained through filtering).
- it can be used for initialization: Figure 5.14 shows images of the conversion from an implicit representation of the object, to an explicit model.

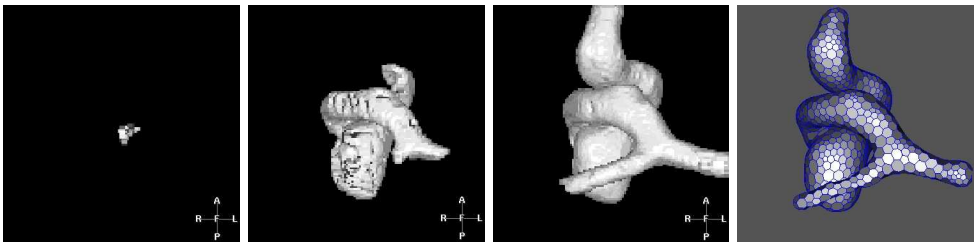


**Figure 5.14. Combining Fast-Marching with Explicit Methods :** the first three images are samples of the segmentation process described in section 5.3; the right image is the conversion of the implicitly defined model into a simplex mesh.

**Drawbacks**

- Monotonicity of the speed (can only propagate in one direction);
- No Curvature term in the Energy of the contour to be extracted;
- No stopping criterion (Speed always strictly positive/negative).

### 5.4.2 Advantages and Drawbacks of *Level-Sets*

*Level-Sets* implementation allows to evolve embedded curves with wide variety of force models, and lead to high-accuracy segmentations with sub-pixel precision. But they often use time-consuming iterative approaches. They have always been criticized because of their computational cost. For instance being used for the segmentation of the cortex (see for example *Zeng et al.* [195]), they provide very satisfying results, but several hours of computation are needed. Several implementations try to override this problem.

The speed of the algorithm has been increased with the use of semi-implicit discretization schemes by *Goldenberg et al.* [65, 67]. This method is based on the implementation of a semi-implicit discretization scheme originally applied to non-linear diffusion filtering [182]. Only drawback is the assumption that the slope of the *Level-Sets* guarantees $\|\nabla\phi\| = 1$ across iterations, with an evolving equation based on the formulation of [23]. This condition is not met with the *Level-Sets* formulation, but another model developed in [69] overcomes this drawback.

An interesting algorithm was derived in [137]: the *Hermes* algorithm is based on the idea to propagate the pixel that evolves faster at each step, thus combining the *Narrow-Band* and *Fast-Marching* approach, with propagation over a relatively small window. However, the proposed algorithm does not solve exactly the given partial differential equation since the evolution is computed locally, and this acceleration step is not valid in an homogeneous media, where the whole curve is propagating with the same speed (as shown for a pure advection flow in figure 4.6).

### 5.4.3 Combining two complementary methods

In the domain of medical image analysis, time-consuming algorithms are synonymous with non-interactive methods, and are therefore limited to a very small number of specific applications. In others words, the computation times of level sets are an obstacle to a wider use of these methods. The strategy we used is an original approach which tries to go beyond this classical barrier.

Both methods have a common advantage : they have no constraints or hypothesis on topology, which may change during convergence.

Our segmentation strategy will consist in using *Fast-Marching* method as a pre-segmentation tool, and then refine the segmentation with a level set method. This approach combines the advantages of both methods: the fast-marching pre-segmentation is rough but quick, and the level set needs only a few iterations to produce the final, highly accurate segmentation.

By combining fast-marching and level sets, we build a tool which is able to produce highly accurate segmentations of topologically and geometrically complex structures in minutes where level sets alone took hours. The framework is the following

*Fast-Marching* will provide a fast and rough initialization, near the solution. As a second process it will give a statistical study of the pre-segmentation (for the level-sets forces): looking at the local distribution of the different regions extracted with the competitive front, we are able to give the initial values of the region descriptors. Computing the mean and variance of different regions, we can skip the **MDL** statistical study and assume that those values are the original first and second moments of the Gaussian probabilities in section 5.2. And *Level-Sets* will start from the segmentation step achieved by *Fast-Marching* . In a few iterations, they will converge with more complex external forces, including region-based terms as in sections 4.4 and 5.2. Their particular formulation will lead to an accurate position of the sub-pixel iso-surface, that enhances visualization and measurements of pathologies.

# Chapter 6

# Applications of *Fast-Marching* and *Level-Sets* shape extraction framework

**Résumé** — Dans ce chapitre, nous avons étudié deux problèmes où la mesure et la visualisation du résultat de la segmentation sont primordiales.
Tout d'abord, on a étudié les anévrismes du cerveau, qui sont des gonflements des artères, qui peuvent mener à une hemorragie cérébrale et plonger le patient dans le coma. Dans ce cas, c'est la précision et la rapidité de la segmentation qui sont des éléments fondamentaux pour préparer une intervention chirurgicale.
Le second problème est celui des polypes du colon. Dans le cas de la colonoscopie virtuelle de la section 3.1, l'utilisateur regarde l'intérieur du colon, et la détection des polypes repose entièrement sur ses indications. Nous avons utilisé notre méthode de segmentation pour automatiser et rendre robuste cette tâche de détection en améliorant la visualisation.

**Abstract** — In this chapter, we have studied two different problems related to typical pathologies, where measurements and visualization are the main objectives. First problem is the segmentation of cerebral aneurysm which are dilation of the brain vessels that may burst and lead to coma. Accuracy and speed of the segmentation process are needed in order to prepare interventional treatment. We therefore applied successfully our methods to segment and extract a shape representation of cerebral aneurysms.
Second problem is the already studied colon polyps visualization. In virtual colonoscopy, as developed in section 3.1, the user observe the interior of the colon, and detection is based on its indications. We want to automate this task, and we give preliminary results on an interesting visualization mode that could lead to this target.

## 6.1 Segmentation of cerebral aneurysms

**What is an aneurysm?**
An aneurysm is an abnormal dilatation involving the wall of an artery, a thick-walled blood vessels that carry blood pumped from the heart, under high pressure. Aneurysms can develop on any artery within the body. Common arteries where aneurysms are found include the aorta, the popliteal artery (behind the knee) and brain arteries. We focus on the brain aneurysms (see figure 6.3-right). A brain aneurysm begins as a small thinned area at the wall of an artery at the base of the brain (see figure 6.1). Over time the blood flow pounds against the thinned portion

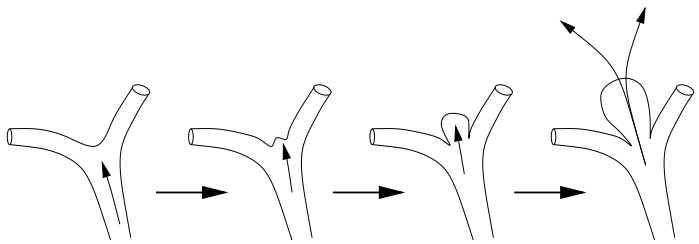


**Figure 6.1. Development and rupture of a brain aneurysm**: The blood flow, represented by an arrow, stresses an area of "potential" weakness at the branching point of an artery. Over time, this point of weakness dilates into an aneurysm. The blood enters the aneurysm itself and finally escapes from a rupture point at the top of the aneurysm.

of the wall, and eventually it starts to dilate, inflating like a balloon, creating the aneurysm, and as it grows, the wall of the artery gets thinner and thinner, until it ruptures.
**What causes an aneurysm to form?**
This is still an open question. Aneurysms arise at an area where the wall of an artery is thin. Most arteries in the body have walls with three layers, and brain arteries have segments where one of the layer is absent, which can contribute to the problem. An aneurysm may be caused or aggravated by disease such as hypertension, because it results in high blood pressure, or may be caused by any disease which affects the walls of the arteries (even smoking).
**What dangers do aneurysms present?**
The danger from an aneurysm is that it will continue to bulge and may burst. When an aneurysm in a large blood vessel or in the heart bursts, a person could bleed to death. When an aneurysm bursts in the brain, a stroke (brain attack) can result.
**How are cerebral aneurysms diagnosed?**
Biplane angiography has become a standard imaging procedure for the treatment of cerebral aneurysms. Recently, there has been a lot of interest in 3D visualization of intracranial vessels in interventional neuroradiology [95]. A clinical application of 3D-Rotational Angiography (**3D-RA** ) has been developed by Philips, using a standard angiographic system, with a C-arm that performs a rotational angiographic acquisition around the patient, and provide accurate 3D reconstruction [45]. With

classical rendering techniques, it enables the clinician to observe for example the relationship of a parent vessel with the neck of an aneurysm, in the brain vessels. The availability of three-dimensional information during the intervention increase the possibilities towards a more accurate and time efficient endovascular treatment. With volume rendering tools, the clinician is able to see structures from any angle. But still, it relies on threshold-based visualizations techniques.

**How are cerebral aneurysms treated?**
There are two ways to treat an aneurysm: First one is surgical treatment, where the clinician places the patient under general anesthesia. A window is opened in the skull bone of the patient head. When the aneurysm is in view, it is separated from the surroundings structures by dissection, and a metal clip is closed across the base of the aneurysm. Therefore blood no longer flow into the aneurysm (see figure 6.2-middle). Second way is to do endovascular coiling: this technique consists of filling the

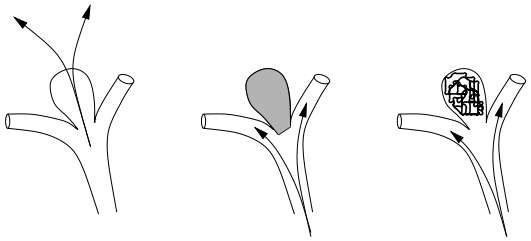


**Figure 6.2. Treatment of a brain aneurysm**: Left image shows an aneurysm at a branch point; middle image illustrates the placement of a clip across the base of an aneurysm correctly placed; right image illustrates the packing of the same aneurysms with coils.

aneurysm from the inside with a long length of fine platinum wire that coils around and around inside the aneurysm, until the blood flow stops entering the aneurysm (see figure 6.2-right). In order to reach the aneurysm, the clinician puts a fine catheter through the artery, and advances it into the head until it reaches the aneurysm. Then the wire is passed up into the catheter and into the aneurysm, until it is full of wire. This treatment has its own risks and complications, since it is still a very invasive technique. The aneurysm can burst during the treatment. In order to perform this examination in the best condition, the clinician needs to output from the angiography an accurate model of the aneurysm. The surface of the object segmented can be used to do measurements, but also flow simulations in order to determine critical points of possible rupture.

We have applied the segmentation models used in section 5.4, in order to increase the accuracy of the visualization of the pathologies. Increasing accuracy and speed of the computations is important for a system in an interventional environment. The 3D shape information will enhance analysis of the structures, measure of the vascular anatomy, and preparation of the treatment.

### 6.1.1 Description of the acquisition system

The system uses a standard angiographic system that is equipped with an image intensifier on a C-arm which performs a rotational angiographic acquisition over a range of $180^o$ (see figure 6.3-middle), scanning 100 projection images. The clinician injects

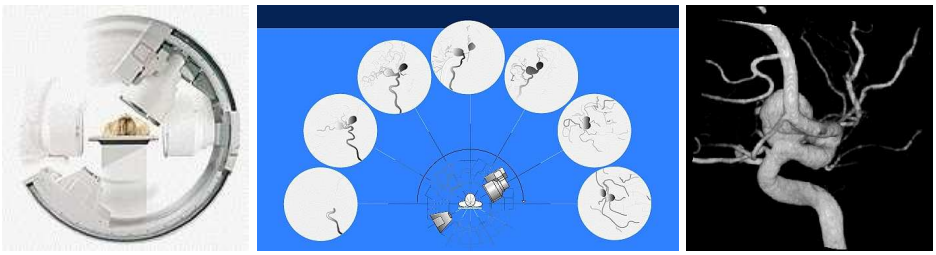


**Figure 6.3. 3D Rotational Angiography (3D-RA) System**: Rotating around the patient, as shown in right image, the system acquire 100 projections along the half circular trajectory, as shown on the middle image; right image is a threshold-based volume rendering of a cerebral aneurysm computed from the 3D dataset reconstructed.

contrast product in the vessel during the acquisition (300 mg/ml iodinated contrast agent at a flow rate of 4-5 ml/s), to fill the pathology of interest during the scan. Acquired scans are then transfered to a workstation, and are automatically corrected for the image intensifier distortion, according to an initial performed calibration step. The reconstruction of the rotational images into a volume is performed by a modified version of the cone-beam algorithm of *Feldkamp* [51] (because of the half-circular trajectory). The default reconstruction procedure, from acquisition until the 3D volume creation takes approximately 6 minutes. And the 3D result can be viewed with a real-time volume rendering package (see figure 6.3-right).

### 6.1.2 Application to the segmentation of brain aneurysms

Threshold-based volume rendering is not sufficient to correctly extract valuable information from the dataset acquired (see figure 6.4 a cerebral aneurysm data, and its **MIP** projection image).

On complex objects, the topology will vary according to the selected visualization threshold on the raw volume. Some vessels will merge, and others split. It appears unreliable to visualize raw reconstructed volumes in terms of topology and vessel size.

We apply the competitive front method, described in section 5.3 to this kind of dataset, in order to operate a supervised fast pre-segmentation of the dataset. Only requirement is to set a seed region inside the object of interest, and another one in the background. The contrast-filled object being clearly bright on the dataset, it is not difficult at all to select a voxel $v_{in}$ inside the aneurysm, and another voxel $v_{out}$ outside it. Using our formulation of two propagating fronts $\mathcal{F}_{in}$ and $\mathcal{F}_{out}$ with two different initial penalty functions $\mathcal{P}^0_{in} = \mathcal{I}(v_{in})$ and $\mathcal{P}^0_{out} = \mathcal{I}(v_{out})$ lead to the segmentation shown in first row of figure 6.6. When the two fronts collides at time $t$,

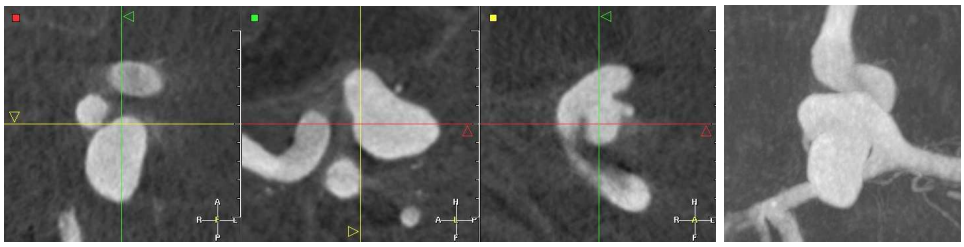


**Figure 6.4. Example of 3D-RA dataset**: The left image represents three orthogonal views of a cerebral aneurysm acquired with the **3D-RA** system; right image is a **MIP** of this dataset.

the new penalty $\mathcal{P}^t_{in}$ and $\mathcal{P}^t_{out}$ are input in our region-based geodesic active contour framework. Knowing the two distributions $(\mu_{in}, \sigma_{in})$ and $(\mu_{out}, \sigma_{out})$, and that the aneurysms are brighter than the background, we initialize the region descriptors with the region following region probabilities:

$$\mathcal{P}_{in}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(\mathcal{I}(x)-\mu_{in}(t))^2}{2\,\sigma^2_{in}(t)}) & \text{if } \mathcal{I}(x) < \mu_{in}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)} & \text{if } \mathcal{I}(x) \geqslant \mu_{in}(t) \end{cases}$$

$$\mathcal{P}_{out}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)} & \text{if } \mathcal{I}(x) < \mu_{out}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)}\ \exp(-\frac{(\mathcal{I}(x)-\mu_{out}(t))^2}{2\,\sigma^2_{out}(t)}) & \text{if } \mathcal{I}(x) \geqslant \mu_{out}(t) \end{cases}$$

The corresponding descriptors are shown in figure 6.5-right.

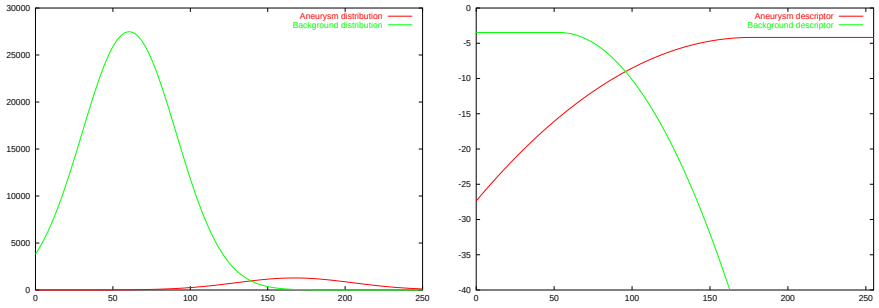



**Figure 6.5. Region descriptors for the 3D-RA image**: Left image shows the estimated Gaussian distribution of the inside and the outside of the cerebral aneurysm segmented by *Fast-Marching* as shown in first raw in figure 6.4; the inside $\rightsquigarrow \mathcal{N}(167, 37)$ and the background $\rightsquigarrow \mathcal{N}(60, 30)$.

Using the region descriptors $k_{in}$ and $k_{out}$ in the model defined in section 5.2, we iterate 20 times from the *Fast-Marching* initialization, to the final segmentation, shown in second row of figure 6.6.

The variability in positioning the initial seed voxels inside and outside the object does not have a clear impact on the final solution, since it provides an initial guess,

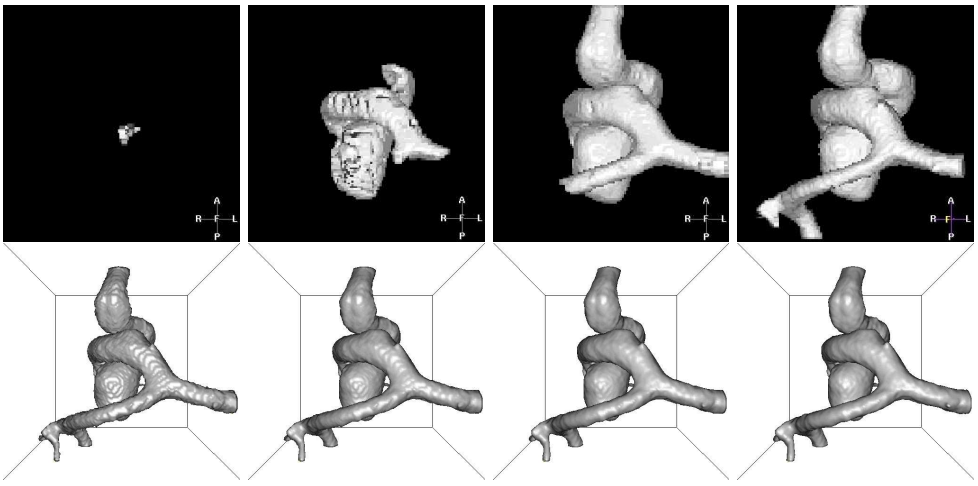


**Figure 6.6. Example of brain vessels segmentation with combination of Fast-Marching and Level-Sets**: First row shows iterations of the segmentation process described in section 5.3; second row shows iterations of the final refinement technique described in section 5.4; parameters for inside object $\alpha = \eta = .8$ and $\zeta = 0.05$.

near the final converged zero level-set, and since it is the initialization of the region-based descriptors. The independence of the region-based level-set framework from the initialization ensures that variability in setting the initial seeds.

The measure of the variability of the positioning of the seed point, whereas it does not have any clear impact on the final result, should be emphasized, since it is the initialization of the region-based descriptors. The variation in the acquisition protocol could optimized the settings of parameters of the boundary based forces, towards region-based forces importance.

And the setting of the seed points could be automated, since the dataset is centered in the volume of interest, and it always intersects the image borders (the intersection could be recognized, being the section of a circular vessel).

The same process, including supervised pre-segmentation and automatic final segmentation was applied to a set of cerebral aneurysms see figure 6.7. The descriptors $k_{in}$ and $k_{out}$ issued from the segmentation model of the cerebral aneurysm of figure 6.4 were used for each dataset. The boundary descriptors attracted the zero-level set of $\phi$ and converges rapidly. Parameterization was the same for each dataset, and stability of the protocol ensured fast convergence to the aneurysms shown in last row of figure 6.6.

The shape extraction of the aneurysm is done in less than 2 minutes on a 300MHz Sun Workstation. Knowing that the reconstruction procedure, from acquisition until the 3D volume creation takes 6 minutes, the added procedure value towards its cost is not important, especially if we consider that on a classical standard commercial PC, this cost could be divided by two. Therefore, this procedure provides all the anatomical information required for the entire course of endovascular treatment of

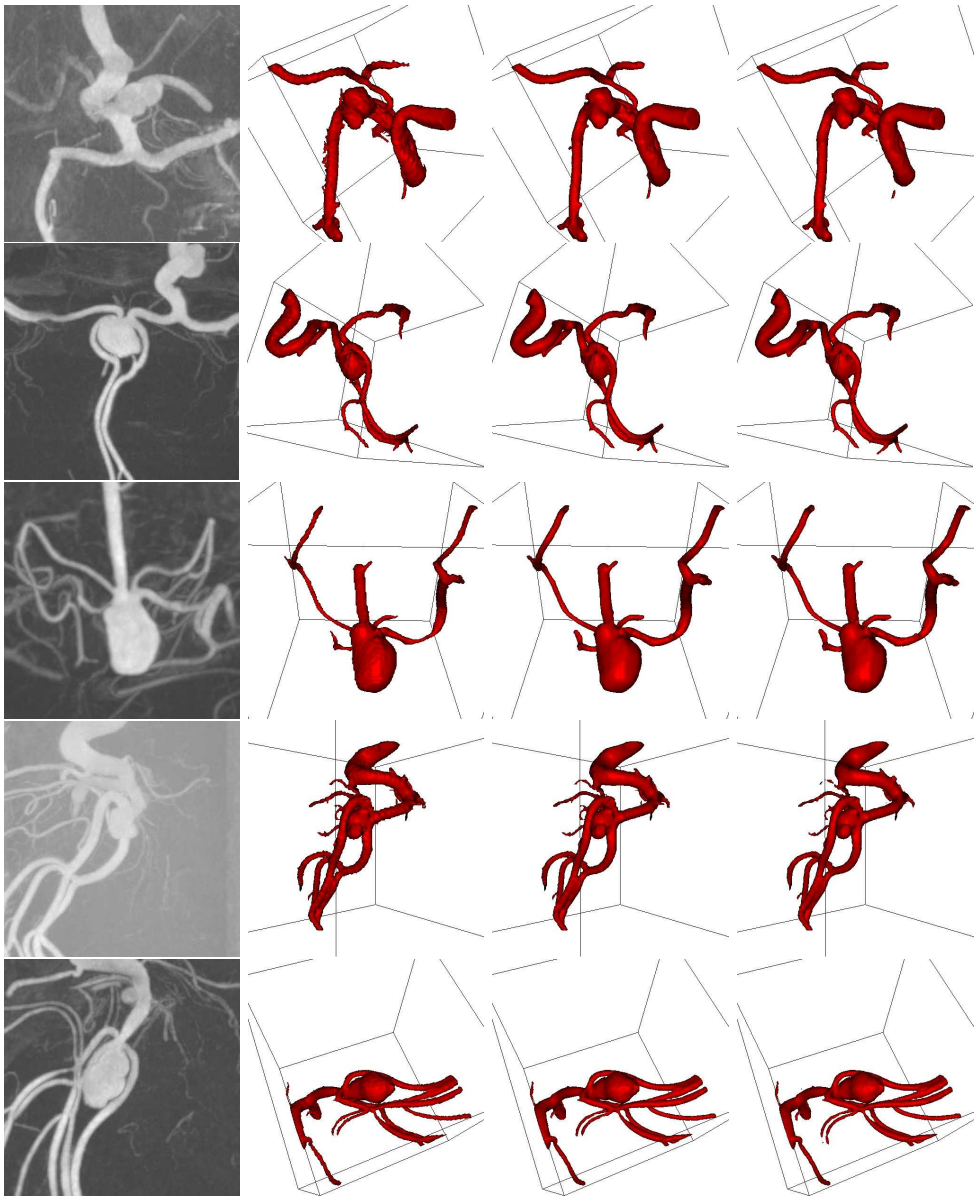


**Figure 6.7. Segmentation of five 3D-RA datasets**: First column contains the **MIP** images of the five data-sets; The surface rendered views show, from left to right in each row, iterations 0, 10 and 20 of the level-set segmentation process.

cerebral aneurysms. Direct benefits for interventional neuroradiology:

1. fast and accurate analysis of the morphology of the aneurysms, including determination of the size and the relationship with the parent vessel, allowing

selection of the appropriate stent size, catheter and guide-wire thickness, or coil length;

2. fast and safe decision regarding the feasibility of endovascular approach.

3. to reduce radiation exposure: **DSA** images are still the gold standard in Angiography during interventional treatment, but when the 3D result is available, less 2D images need to be acquired before, during, and after the treatment;

### 6.1.3 Perspective

The 3D shape representation of the aneurysms creates lots of investigation fields. In particular we could reduce (or suppress) contrast agent injection: the 3D model could be mapped/projected on the 2D image during intervention, by registration techniques, and used as a mask for the treatment (see figure 6.8);

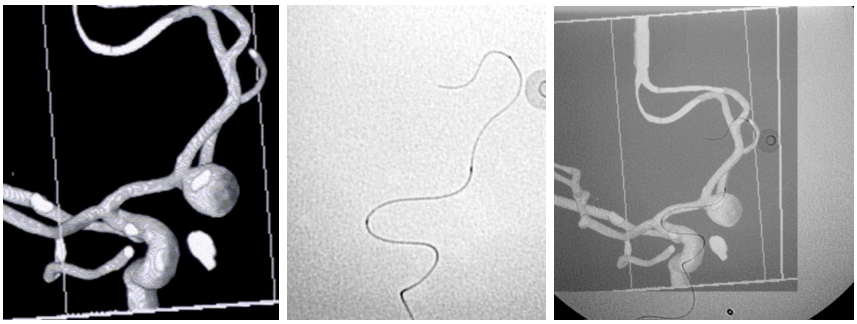


**Figure 6.8. Live road mapping for interventional treatment**: Left image is a 3D threshold based volume rendering; middle image is a 2D X-ray image of a guide and a catheter; right image is the superposition of those 2 images.

Another research direction could be the road-mapping of interventional treatment: the segmentation of the vessel structure enables to extract trajectories inside the vasculature, in order to efficiently reach the aneurysm (for stent placement for example), forecasting the future problems during intervention. Path extraction techniques, studied in chapter 2, could also be useful for extracting the optimal guide shape for an intervention: during intervention, for glue injection, the clinician introduces the catheter in the aneurysm, with the help of the guide. Then when the guide is extracted, the catheter - similar to a spaghetti - can move out the aneurysm, or eventually break it. This dangerous step can be avoided if the catheter has been optimally shaped using a trajectory artificially extracted in the 3D model.

Finally, the explicit surface extracted from the *Level-Sets* representation (with the *Marching-Cubes* algorithm) is a mesh that enables modeling of the blood flow inside the aneurysm. This model can help in preventing bursting of the aneurysms, but could be also very useful in following-up the result of an intervention.

## 6.2 Detection of Colon Polyps

This problem was already mentioned in section 3.1, concerning the path extraction tool developed specifically for virtual endoscopy. As said, colorectal cancer represents the third most frequently diagnosed cancer worldwide. If we consider malignant tumor, the yearly incidence of colorectal cancer probably approaches 160,000 cases [99]. This disease begins in the cells that line the colon, as polyps.

**What is a Colon Polyp?**

A polyp is a growth that occurs in the colon and other organs. These growths, or fleshy tumors, are shaped like a mushroom or a dome-like button, and occur on the inside lining of the colon.

**What dangers do polyps present?**

Colon polyps start out as benign tumors but in time may become malignant. The larger the polyp, the more likely it is to contain cancer cells.

**Why do Colon Polyps and Cancer Form?**

A great deal is known about why and how polyps form. There now is strong medical evidence that there are abnormal genes for colon polyps and cancer that can be passed from parent to child. Diet and foods may also be very important.

**How are Colon Polyps diagnosed?**

Importantly, colon cancer is one of the most curable forms of cancer. When detected early, more than 90 percent of patients can be cured. Early detection of colon polyps and cancer is performed usually with

1. study of the patient's medical history for identification of risk factors;
2. stool examination to detect occult blood from Colon cancers and large polyps;
3. visual examination of the lower colon using a lighted, flexible endoscope;
4. colonoscopy of the entire 5-6 foot long colon, under sedation;
5. x-ray exam (Barium Enema) which outlines the shadows of polyps and cancer;
6. virtual colonoscopy (already developed in section 3.1).

But still, even the *Virtual Endoscopy* relies on the user observation, for the detection, during visualization, of possible polyp existence. We already mentioned a possible unfolded view of the interior of the colon (see figure 3.21) that enables to see in all directions while traveling through the colon, but inspection remains a supervised process that rely on possible miss of hidden regions, from the camera point of view. Last drawback of endoscopy is that it relies on the choice of an opacity threshold input in the volume-rendering tool. The choice of this threshold critically constrain the position of the colon surface, thus the clinical validity of the observation.

### 6.2.1 Segmentation of the colon surface

We propose in the following to adapt the method developed in previous chapter, and already applied to cerebral aneurysm segmentation, to develop a initial framework of

semi-automatic polyp extraction. We further explore possibilities of detection with visualization techniques, using the curvature information of the object surface.

Classical CT scanner are generally very large. Instead of treating entire images, we used small volumes of interest which were selected by specialized physicians because of the presence of a particular pathology.

Before acquisition, the patient goes through a particular preparation during which the colon is emptied as much as possible. During the scan it is distended by inflating room air. The resulting image intensity in the colon lumen is rather uniform and lower than in the rest of the image, with a relatively good contrast. Therefore, the critical step of the segmentation process is the variability of the topology and geometry of the pathological structures.

Since the contrast is really important, as shown in figure 6.9-(a), it is a very easy task to set a seed point inside the colon, and another outside. Thus, supervised seg-

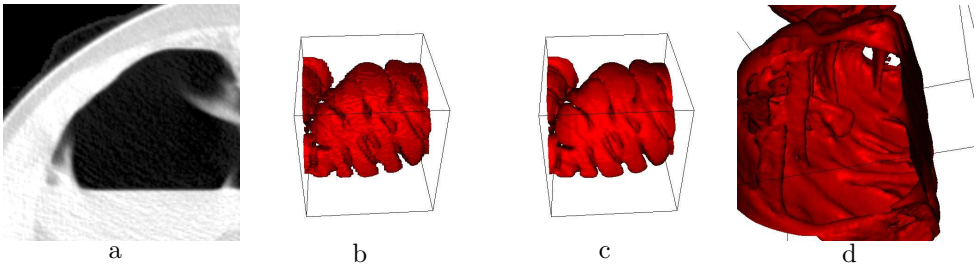
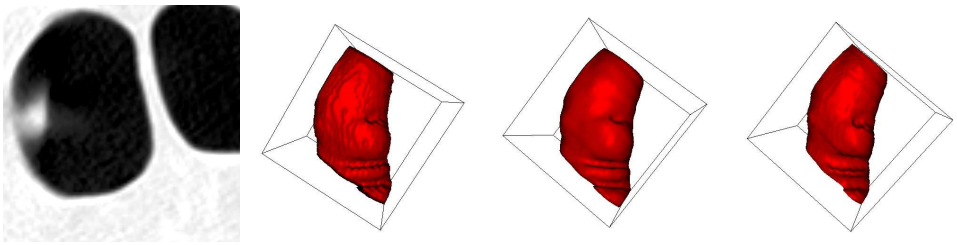



**Figure 6.9. Example of polyps Segmentation**: image (a) is a slice of a volume of interest (VOI) of the 3D CT scanner of the colon studied; image (b) shows the resulting pre-segmentation obtained with the *Fast-Marching* competitive version; image (c) is the result of the region-based *Level-Sets* at convergence, after 20 iterations; image (d) is an endoluminal view of the same segmented object, which emphasizes the polyps that grows on a fold of the colon surface.

mentation with front competition, using the *Fast-Marching* as detailed in section 5.3, can be easily achieved, as shown in figure 6.9-(b). Using the descriptors output by the pre-segmentation process, we initialize our *Level-Sets* with sigmoidal region-based forces. The justification of the use of those forces is the following: Pathological cases can arise, as shown in figure 6.10 In this application, the use of sigmoidal region-based forces is interesting because, due to the topology of the colon, that intersects several times the same volume of interest, it is possible to obtain disconnected parts of the colon in the same volume. Pre-segmentation being based on the setting of an interior seed point and an exterior seed point will lead to a binary image. This means that portions of the colon are probably included in the background. Therefore the statistical study of the background grey-levels will lead, with the Gaussian descriptors, to a background that have a large variance (see figure 6.11-left), whereas local histogram in the colon, due to the contrast, will give a small variance. Values for the example shown in figure 6.10-a, are $(\mu_{in} = 40, \sigma_{in} = 23)$ for the colon, and $(\mu_{out} = 620, \sigma_{out} = 385)$ for the background. Unless the user finds all disconnected parts of the colon, it is easier to use the *choice* of a darker region for the colon, and a brighter one, for the background, since evolution of the Gaussian model could lead to



a b c d

**Figure 6.10. Segmentation tests with different descriptors**: image a is the underlying dataset; image b is the initialization with the *Fast-Marching* algorithm; image c is the resulting segmentation after 100 iterations of the *Level-Sets* sigmoid region based forces; image d is the similar segmentation with Gaussian region based forces.

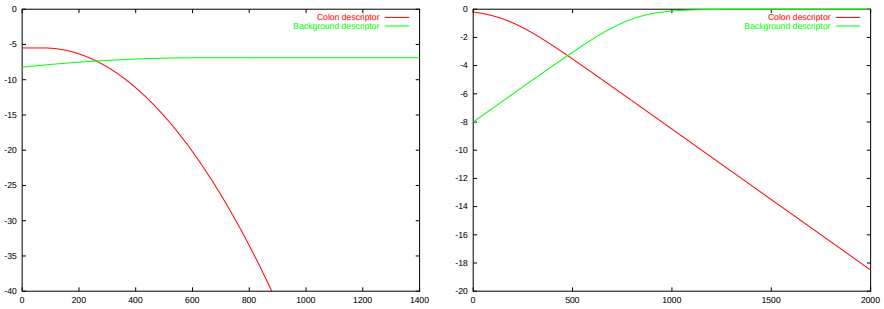



**Figure 6.11. Region descriptors for the colon polyps**: The left image represents the thresholded Gaussian descriptors for the colon polyps of figure 6.9; right image corresponds to the sigmoid descriptors of the same dataset.

unexpected results. In figure 6.10-4, the colon surface is flattened across iterations, because the variance of the background is higher. In figure 6.12, we display the result of the application of the same framework, using the same parameters than for example 6.9.

### 6.2.2 Visualization of the colon polyps

Colon polyps appear as convex regions in the lumen surface, in intraluminal 3D views (see segmentation results in figure 6.12). We tried to enhanced these suspect regions using a color information on the surface.

The specific shape of the colon polyps settle the use of the curvature information, mapped on the surface of the object, using an adequate color-map to highlight the cups. This technique has been already used in the surface of a segmented cortex, by *Zengh et al.* [195], using a measure defined originally by *Koenderink et al.* [94]. Using the values of $\tilde{\phi}$, $\phi(\cdot, t)$ at convergence, we know the expressions of the mean curvature $\kappa_M$ and the Gaussian curvature $\kappa_G$ for a surface propagating in three space

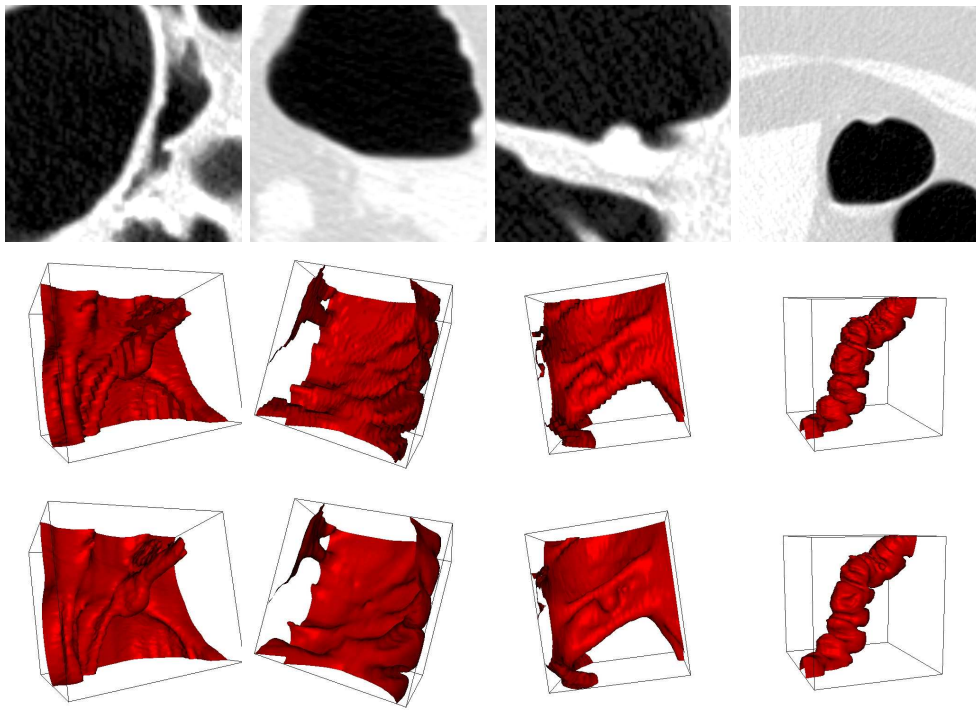


**Figure 6.12. Polyps segmentation**: First row: the four datasets used for segmentation; second row: the respective initializations given by the Fast-Marching competition algorithm described in section 5.3; third row: visualization after 20 iterations with region-based forces of the respective zero level-set.

dimension, in terms of the level-set function $\tilde{\phi}$. They can be easily computed using formulations given by *Sethian* [163]:

$$\kappa_M = \nabla \cdot \frac{\nabla \tilde{\phi}}{|\nabla \tilde{\phi}|} = \frac{\left\{ \begin{array}{c} (\tilde{\phi}_{yy} + \tilde{\phi}_{zz})\tilde{\phi}_x^2 + (\tilde{\phi}_{xx} + \tilde{\phi}_{zz})\tilde{\phi}_y^2 + (\tilde{\phi}_{yy} + \tilde{\phi}_{xx})\tilde{\phi}_z^2 \\ -2\tilde{\phi}_x\tilde{\phi}_y\tilde{\phi}_{xy} - 2\tilde{\phi}_x\tilde{\phi}_z\tilde{\phi}_{xz} - 2\tilde{\phi}_y\tilde{\phi}_z\tilde{\phi}_{yz} \end{array} \right\}}{(\tilde{\phi}_x^2 + \tilde{\phi}_y^2 + \tilde{\phi}_z^2)^{3/2}} \tag{6.1}$$

$$\kappa_G = \frac{\left\{ \begin{array}{c} \tilde{\phi}_x^2(\tilde{\phi}_{yy}\tilde{\phi}_{zz} - \tilde{\phi}_{yz}^2) + \tilde{\phi}_y^2(\tilde{\phi}_{xx}\tilde{\phi}_{zz} - \tilde{\phi}_{xz}^2) + \tilde{\phi}_z^2(\tilde{\phi}_{xx}\tilde{\phi}_{yy} - \tilde{\phi}_{xy}^2) \\ +2[\tilde{\phi}_x\tilde{\phi}_y(\tilde{\phi}_{xz}\tilde{\phi}_{yz} - \tilde{\phi}_{xy}\tilde{\phi}_{zz}) + \tilde{\phi}_y\tilde{\phi}_z(\tilde{\phi}_{xy}\tilde{\phi}_{xz} - \tilde{\phi}_{yz}\tilde{\phi}_{xx}) \\ +\tilde{\phi}_x\tilde{\phi}_z(\tilde{\phi}_{xy}\tilde{\phi}_{yz} - \tilde{\phi}_{xz}\tilde{\phi}_{yy})] \end{array} \right\}}{(\tilde{\phi}_x^2 + \tilde{\phi}_y^2 + \tilde{\phi}_z^2)^2} \tag{6.2}$$

We can use the scalars obtained and attach them to the vertices of the triangulated surface extracted by the *Marching-Cubes* (see figure 6.13-(b) for the surface extracted). However those values do not give valuable visible information that discriminates the structures we are looking for. We know that those convex structures have the particularity to have high principle curvatures $\kappa_1$ and $\kappa_2$. Knowing that

$\kappa_M = \kappa_1 + \kappa_2$ and $\kappa_G = \kappa_1 \times \kappa_2$, we deduce immediately the value of the principle curvatures.

We can map the maximum of $\kappa_1$ and $\kappa_2$ on the surface of the extracted object, as shown in figure 6.13-(c), but it does not give a clear view of the polyps.

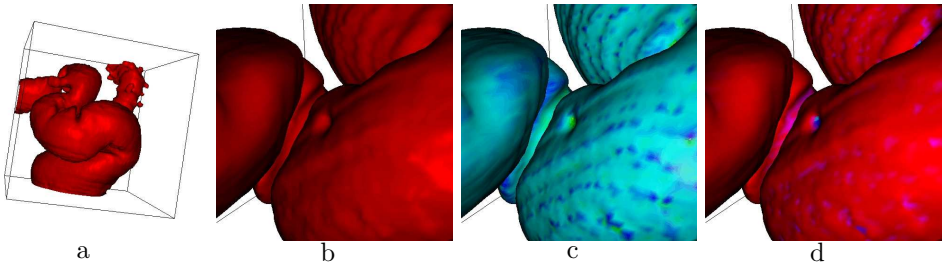



**Figure 6.13. Example of polyps visualization**: First image: the segmentation obtained by combining Fast-Marching and Level-Sets; second image: a zoom on a region of the colon volume; third image: the mapping of the function of the principle curvatures computed; fourth image: the threshold of this texture which highlights the polyps.

Furthermore, having in mind that we are looking for regions with negative curvatures, we apply the following equation

$$f(\kappa_1, \kappa_2) = \min(\max(\kappa_1, \kappa_2), 0) \tag{6.3}$$

which is interpolated at the vertices of the triangulated surface as shown in figure 6.13-(d). Only the regions where the two principle curvatures are negatives (*cups*) have negative $f$ values, others get null values.

### 6.2.3 Perspectives

Unfortunately, other non-pathological regions are enhanced. The last row in figure 6.14 displays the result of this curvature mapping for four different datasets. Segmentation step was achieved using the same parameters for each datasets, and the curvature mapping is done with the same color-map. On several datasets, this mapping highlights other non-pathological regions: folds can be highlighted because of the sign of their principle curvatures. However our approach might be consider as a valuable start for the automatic detection of polyps, and currently viewed as an assisting tool for their visualization. The polyps are emphasized, and discriminated from the whole surface. Therefore, segmentation and visualization is achieved with a simple and fast process, leading to a pre-detection of the polyps which can already be used by any clinician.

In conclusion, the use of this kind of curvature filter outputs information relative to small and spherical polyps. Those polyps can grow and develop malignant tumor with non-smooth shapes where the curvature information is not suitable. Our tool finds its application in the early detection of the small polyps. The high precision of the implicit level-set representation of the surface obtained through the segmentation

process, enables to map on the surface informations for small objects like polyps. Our curvature measure is dedicated to this visualization.

Having in mind the settle of a non-supervised method of polyps detection, next step is recognition: Other non-pathological objects that are pre-detected can be discriminated with classification of the shapes of the connected components of the subset of the surface which corresponds to negative values of our function in equation (6.3). In last row of figure 6.14, we can imagine that we are able to unfold the surface of our colon, keeping the curvature information mapped on the explicit representation extracted at convergence of $\phi$. This technique has been developed for level-sets techniques by *Hermosillo et al.* [76] using an explicit representation of the surface, with the curvature information mapped onto, which matches the implicit representation during its deformation (in this case, the mean curvature flow to unfold the brain surface, with a constraint on volume conservation). It was also developed in a different manner by *Bertalmio et al.* [12], where the region of interest is tracked as the intersection of two level-sets. Their application is to unfold the cortex in order to see the cerebral activity (mapped onto the surface with a given colormap) in the hidden sulci. Another possible technique is surface warping: using a warping based on registration methods , we can flatten the colon underlying triangulated surface of the zero level-set, into the plane, using a conformal mapping method, as in [73], or another mapping which preserve areas (as done in [74]).

Another possible development is the correct choice for the image scale. The image scale is an important parameter for the aspect of the surfaces and the regularity of the curvatures, but of course large smoothing factors can change the topology of the resulting segmented surfaces. The use of curvature flows seem unadapted as well [44]. A flow based on the intrinsic Laplacian of the mean curvature (see *Chopp and Sethian* [28]) might be an interesting tool to experiment in the future, but the numerical issues related to this flow are still to be explored, and no numerical scheme is currently available.

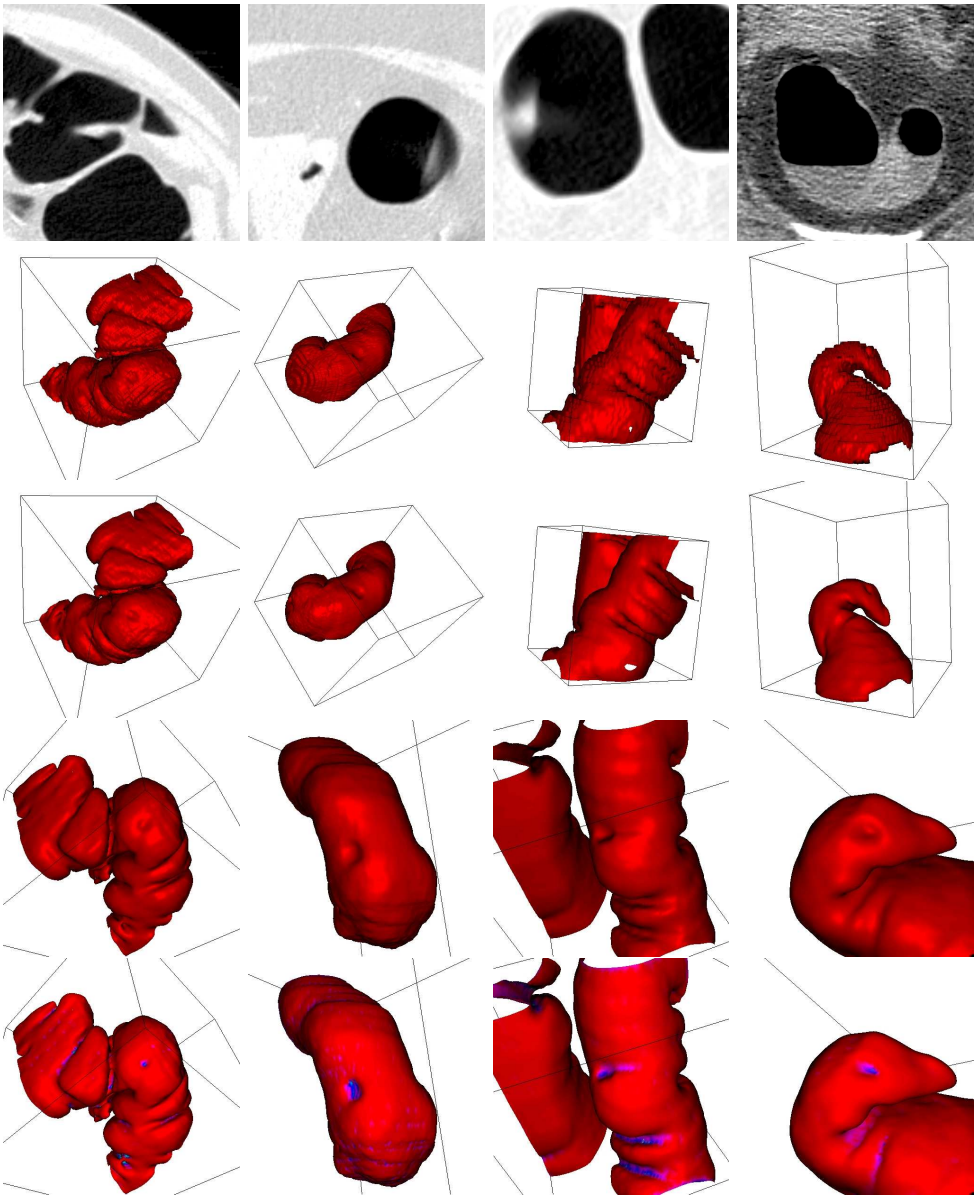
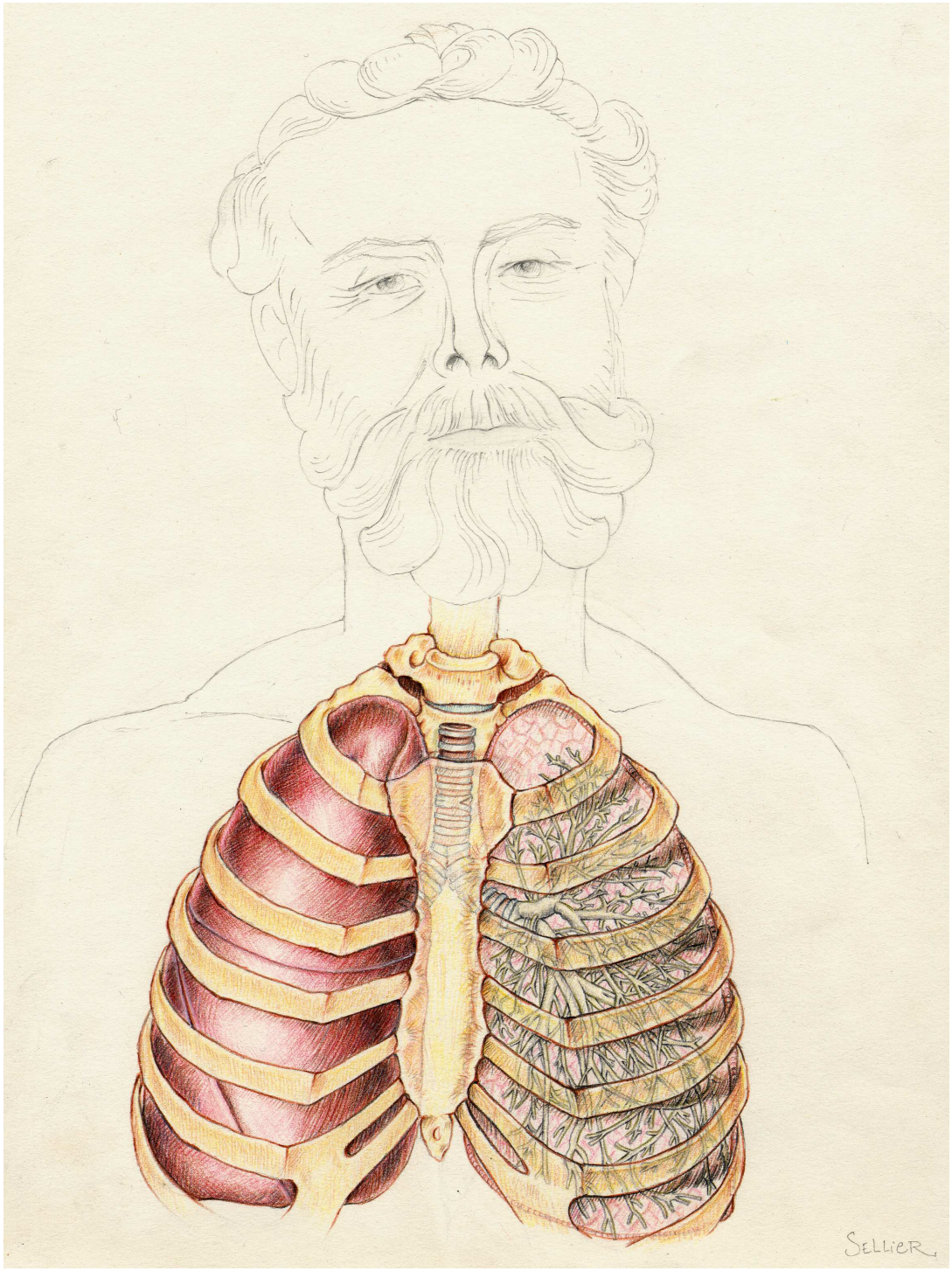


**Figure 6.14. Polyps Visualization**: First row: the four datasets used for segmentation and visualization; second row: the respective initializations given by the Fast-Marching competition algorithm described in section 5.3; third row: visualization after 20 iterations with region-based forces of the respective zero level-set; fourth row: another point of view for visualizing the polyps; fifth row: texture mapping with the curvature information.

# III

# Visualization and quantification of 3D tree-shaped anatomical objects


SELLIER

# Chapter 7

# Visualization and Quantification Tools

**Résumé** — Dans ce chapitre, nous présentons différents outils nécessaires pour la visualisation et la quantification de pathologies dans les objets segmentés. Comme nous l'avons déjà fait dans le cadre d'applications comme l'extraction de surfaces d'anévrismes dans des images 3DRA, ou bien la visualisation des polypes à la surface du colon, dans le chapitre 6, notre outil de segmentation de surface basé sur les ensembles de niveaux et le *Fast-Marching* permet d'extraire et de représenter des objets en 3D.
En premier lieu nous parlons brièvement des problèmes soulevés par la visualisation de surfaces implicites en 3D, et en particulier des spécificités des surfaces définies par les *Level-Sets* dans la section 7.1.
En s'appuyant sur un ensemble de trajectoires qui décrivent nos surfaces - comme le squelette dans le cas de structures arborescentes - nous développons ici des outils de mesure et d'observation des pathologies. Notamment, nous nous intéressons à la mesure du volume en section 7.1, et à la mesure de sections de nos objets segmentés en section 7.2. Ces outils seront utilisés dans toute la suite de cette partie.

**Abstract** — In this chapter we introduce the different necessary tools for visualization and quantification of our segmented objects. The final result of a segmentation, given by our framework as done in chapter 6 for different applications, can lead to visualization and measures on the global object.
We first briefly present the problems of visualization of an implicit surface in 3D, and more precisely the specific drawbacks of the *Level-Sets* representation in section 7.1. Assuming that we can extract a whole set of trajectories in a tree-shaped object, we present the different tools that will measure the pathologies, on the basis of those trajectories. Important measurements include: volume measurements, as explained in section 7.1, and objects cross-section measurements, as detailed in section 7.2. Those tools are useful for the framework developed in the following chapters.

## 7.1 Visualization of 3D segmentation

In the domain of medical image analysis, the segmentation tools we developed are essentially interesting when applied to 3D images. This is the reason why the implementation we build is designed for this image dimensionality, and that our visualization efforts were mainly directed to 3D techniques.

In this section the reader will first find a short presentation of the basic notions of virtual reality needed to understand the content of our work. They are grouped together in the first subsection, which can be skipped by the readers who are already familiar with them.

### 7.1.1 Virtual reality notions

Classically, the basic techniques for computer graphics of virtual reality rely on the computation of *renderings* of virtual 3D *scenes*. A scene is composed of virtual *actors*, *lights* and a *camera*.

#### What are actors ?

The term *actor* covers everything that might be seen when properly enlighten. For instance, in a virtual reality model of a house, each piece of furniture would be modeled by a specific actor, and so would be the floors, walls, stairs, etc.

Traditionally, the shape of a 3D actor is explicitly modeled by a set of graphic primitives: points, lines and surface patches. In recent and advanced models, the surface of an actor is sometimes modeled using implicit functions.

According to the complexity of the modelization, the rendered appearance of the surface of an actor can depend on many and various parameters:

- the position and orientation of the camera relatively to the actor surface;
- the properties of the surface which are taken into account by the *illumination model*;
- the positions, orientations, colors and attenuation factors of the lights, which can be at finite distance (punctual lights) or infinite distance;
- the positions and orientations of the other actors which may cause occlusions, projected shades, or even reflect light sources in advanced models.

#### What is an illumination model ?

The illumination model is the set of equations used to compute the color and brightness of a point on the surface of an actor according to:

- the angles of incidence, intensities and colors of the incident rays of light;
- the modeled properties of the surface;
- the angle of the departing ray of light.

The surface properties usually include colors, an opacity factor, reflectance, a specular power parameter, etc...

In addition to the illumination model, a *shading* model can be used to avoid the faceted aspect of polygonal surfaces. The most popular shading models are Gouraud and Phong shadings, although Phong shading is rarely used because of its computational cost.

#### What is the exact role of the camera ?

The camera plays the same role as in the making of a movie: the rays of light which encounter the objective determine the rendered (i.e. virtually acquired) 2D image. The usual parameters of a camera are its position, orientation, and two of the following parameters: view angle, focal distance and image size. The rendered 2D image is a projection of the illuminated actors in the focal plane of the camera.

#### Object-based and image-based rendering

The construction of the 2D image acquired by the camera may be *object-based* or *image-based*. In the case of object-based rendering, the actors are rendered one by one by applying the illumination model and the projection equations to the graphic primitives they are composed of. The occlusions are generally dealt with using a so-called *Z-buffering* technique: the final image is the result of the superposition of layers which correspond to different depths (Z-coordinate) in the scene. The points that are the closest to the camera are visible, others are more or less occluded according to the opacity of the points that are in front of them.

Object-based rendering is not a recent technique, but it is fast, rather simple, and can be accelerated by specialized hardware devices. For example, OpenGL hardware implementations make interactive renderings of simple scenes possible even on a low end PC. The main drawbacks of object-based rendering are that photo-realistic images are difficult to achieve, especially in the case of complex scenes, and that multiple reflections are usually not taken into account. Moreover the actors have to be explicitly represented using graphic primitives.

In the case of image-based rendering (or *ray-tracing*) the color and brightness of each point of the rendered image is computed by tracing a ray starting from this point. The illumination model is invoqued when the ray hits an actor, and reflections on several actors are even possible before reaching a light source. In the most advanced computer graphics software products based on ray-tracing, the actors can also have implicit representations.

The images produced by ray-tracing can be of very high quality, but the major drawback of image-based rendering is the computation cost related to the calculation of the rays.

### 7.1.2 Visualization of a level-set

Visualizing a level set is nothing more than visualizing an iso-surface in 3D, or an iso-contour in 2D. More generally the hypersurface which needs to be visualized is

the zero-level set of $\phi(\cdot, t)$, where $t$ is fixed, which is a function defined over the image domain $\Omega \subset \mathbb{R}^3$ in our applications. In figure 7.1-left, the surface of a sphere is implicitly defined by the signed distance to itself.

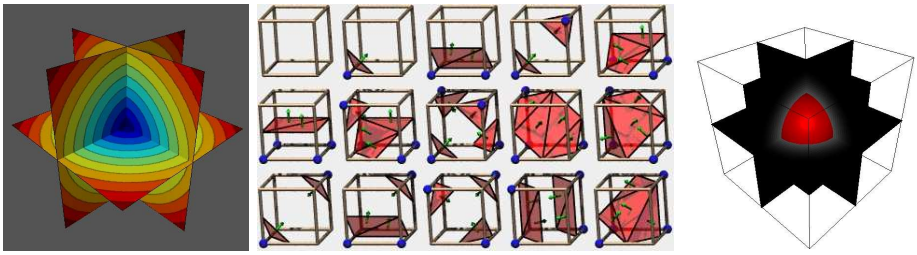


**Figure 7.1. The marching cubes algorithm**: Left image represents the iso-values of the signed distance to a sphere in 3D; middle image represents the different configuration encountered by the *Marching-Cubes* ; right image is a smooth surface rendering of the triangles that approximate the implicitly defined sphere of figure 7.1-left given by the algorithm.

The 3D visualization of the zero level-set surface by object-based rendering algorithms cannot be done directly. An explicit representation of the surface by polygonal graphic primitives has to be computed first.

Several approaches are possible for the computation of a polygonal approximation to an iso-surface. The most popular of all is certainly the *Marching-Cubes* (see [104]), which computes a triangulated surface. In each cube formed by eight contiguous voxel centers, the values of the implicit function at the vertices of the cube are compared to the specified iso-value. The possible configurations are classified (see figure 7.1-middle), and a look-up table is used to quickly give a triangulated approximation of the intersection of the iso-surface with the currently examined cube. All the cubes are examined one by one in a raster-scan "marching" fashion, in opposition to the algorithms which try to "track" the iso-surface.

But sometimes the *Marching-Cubes* algorithm generates triangle sets containing holes, due to ambiguous cases. Many authors have proposed solutions, for example the marching tetrahedra algorithm in [166].

However, we chose the *Marching-Cubes* for reasons of accuracy, reliability, and (above all) simplicity of use since efficient implementations of it are available. It provides an accurate triangulated surface whose precision leads to high-quality renderings, like in the endoscopic images shown in figure 7.2.

### 7.1.3 Problem with the *Marching-Cubes*

A classical evolution equation defined by $\frac{\partial \phi}{\partial t} + V \,.\, \nabla \phi = 0$ makes no distinction between the level sets of $\phi$. They are all attracted by the same asymptotic hypersurface provided that they are sufficiently "close" to it. As a result, $\phi$ gets very steep in its vicinity, which causes the *Marching-Cubes* to give poor and aliased results.

In fact, the level-sets do not remain a distance function in many cases (an exception is a constant advection flow, for example see figure 4.6). This property at

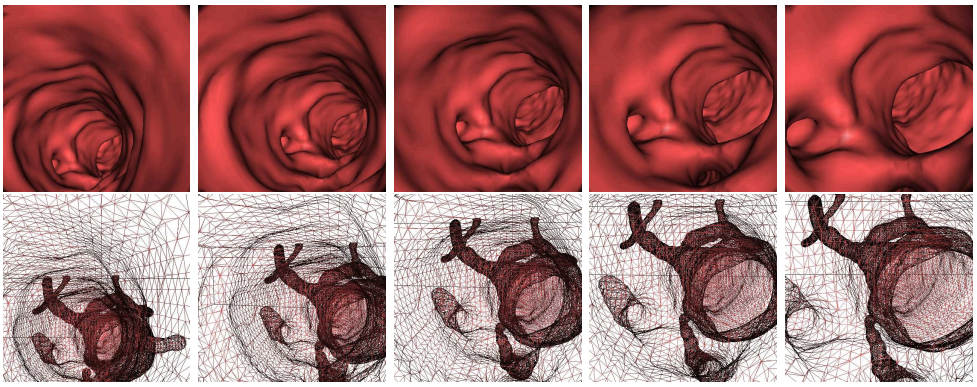


**Figure 7.2. Surface rendering in the aorta**: First row shows frames of an endoscopic movie in an aorta MR dataset. Second row is the wire-frame version of this movie, given by the *Marching-Cubes* where we can see the whole anatomical object and its several branches by transparency

initialization, is lost after several iterations. Figure 7.3 shows two examples: the first one follows a balloon forces, which is positive inside a circle, and negative outside; the second one is a flow composed of a positive balloon force and a boundary based force, which stops the level sets of $\phi$.

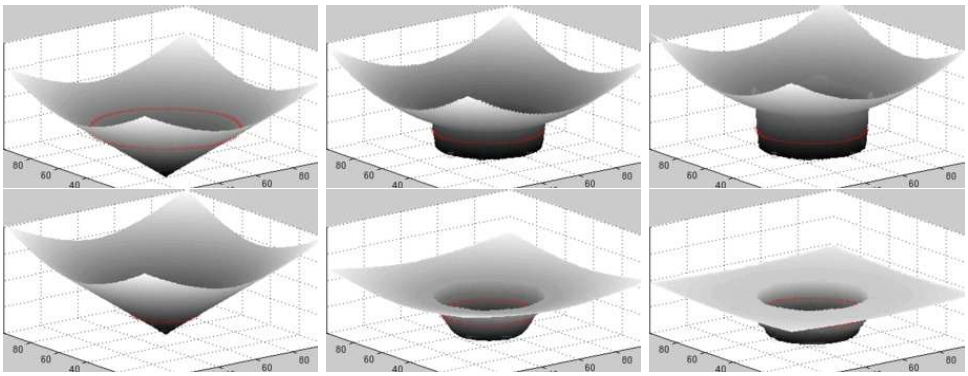


**Figure 7.3. Loosing the distance function when converging**: First row shows consecutive iterations of the level-sets of a geodesic active contour that minimizes the distance to a circle; second row shows consecutive iterations of the level-sets of a geodesic active contour that inflates according to a balloon force with boundary based forces on the same circle.

### 7.1.4 Restoring the distance function

In conclusion, the solution to the classical Hamilton-Jacobi evolution equation proposed in [135] is not a distance function. But this property is the hypothesis of several

numerical techniques to accelerate convergence, like the fast geodesic active contours proposed in [65] and [67]. Moreover, the practical application of the level-set method is plagued with such questions as: when do we have to "reinitialize" the distance function? How do we reinitialize" the distance function. In [163], the author suggests that when the zero-level set evolves in the vicinity of the borders of the narrow-band, the distance to the zero-level set must be re-initialized. For the authors of [68,69], this problem reveals a disagreement between the theory and its implementation, the authors propose an alternative to the use of Hamilton-Jacobi evolution equation which eliminates this contradiction. In order to reach this goal, they look for a function $B : \mathbb{R}^3 \times \mathbb{R}^+ \to \mathbb{R}$ such that $\frac{\partial \phi}{\partial t} = B$ and which satisfies the two constraints

- $\phi$ is a distance function
- $\frac{\partial \phi}{\partial t} = \beta \mathcal{N}$ where $\beta$ is the velocity, and $\mathcal{N}$ the inward unit normal.

Those constraints lead to the new relation $\nabla \phi \cdot \nabla B = 0$. This efficient method increases the computing cost.

Following the author of [84], we include the a restoration force in the Hamilton-Jacobi flows, which not only ensures the evolution of $\phi(0,t)$ given by equation (4.6), but also prevents $\phi(\cdot,t)$ from getting too steep in the vicinity of $\phi(0,t)$. This new partial differential equation is given by:

$$\frac{\partial \phi}{\partial t} + V \,.\, \nabla \phi = \mu \,.\, \mathrm{sgn}_\theta(\phi(x,t)) \,.\, (1 - \|\nabla \phi\|) \tag{7.1}$$

where $V$ is the flow defined by equation (4.6) and where the modified signed function $\mathrm{sgn}_\theta$ is defined by:

$$\mathrm{sgn}_\theta(y) = \begin{cases} -1 & \text{if} \quad y < -\theta \\ \frac{y}{\theta} & \text{if} \quad -\theta \leq y \leq \theta \\ 1 & \text{if} \quad \theta < y \end{cases}$$

The new differential operator introduced in equation (7.1) is inspired from the distance function restoration operator used in [170]. The modification of the sign function avoids the apparition of oscillations during the numerical approximation of equation (7.1), without having to introduce numerical flux or slope bounds. Otherwise, as signaled in [84, page 56], these oscillations are responsible for short but annoying displacements of the zero-level set of $\phi(\cdot,t)$. And the author of [84] proposes to use another scheme, originally presented in [159], which inflates and deflates successively the level-set in order to extract the distance to the zero level-set, without displacing it. We choose not to add another bunch of computations to our method, and decide to use method of [170].

The parameter denoted by $\theta$ can be set to a fixed value (we used $\theta = 10$ in our experiments). The parameter $\mu$ defines the weight of the newly introduced differential operator, and has to be adapted according to the other forces parameters. If $\mu$ is too small, then $\phi(\cdot,t)$ is likely to get too steep for the *Marching-Cubes* to give good results. But too high values of $\mu$ will increase the global CFL number, and thus cause the convergence of $\phi$ to be slower. In practice, it is not difficult to find a good value for $\mu$.

The use of this new equation (7.1) is illustrated in figure 7.4, where the flow drives the zero level-sets to a sphere.

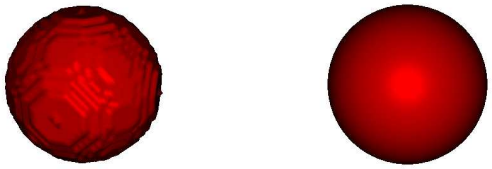


**Figure 7.4. Aliasing when converging**: Left image is the surface extracted at convergence when the level set matches the surface; Right image is the same result including a force to restore the distance function.

### 7.1.5 Volume versus Surface Rendering

*Volume rendering* is an advanced image-based visualization technique based on the integration of a transfer function along rays cast in a dense volume (i.e. a 3D image). The transfer function is generally based on the intensity and gradient of the image, and gives an opacity value for each voxel of the image. Surface (see figure 7.5-left) versus Volume (see figure 7.5-right) rendering is still an open question, and the choice between those two methods depends on the application.

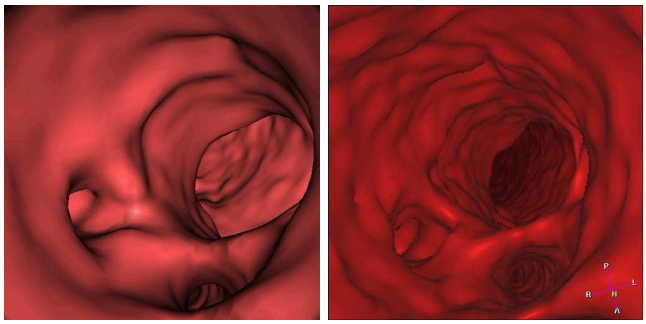


**Figure 7.5. Comparing surface and volume rendering in the aorta**: Left image is a surface rendered view generated with the *Marching-Cubes* on the final segmentation obtained; right image is a threshold based volume rendering view at the same location in the dataset.

With the shape extraction techniques we use, surface rendering has several advantages:

1. with the segmentation framework we have developed, the visualization of the anatomical object with surface based rendering does not need any input, any interaction (unless the color of the surface can be considered as an important parameter);

2. parameterization-free means robustness. Volume-based rendering relies on the critical choice of a suitable transfer function. Surface-based rendering is the direct representation of the surface extracted by the segmentation whereas the volume-based rendering relies on the user perception of the dataset;

3. Surface rendering is fast: when the triangulation has been extracted with the *Marching-Cubes* , endoscopic fly-through, like in figure 7.2 are generated in real-time, and OpenGL hardware implementations, now available on any low end PC, accelerate the computations. The computational cost of volume rendering is very high, and special hardware devices that might overcome this lack of performance are still under development.

Moreover, several artifacts may occur when using *Volume Rendering* on volume data (among them, aliasing, stair-casing and slicing, see [146]).

For all those reasons, we used the surface-based rendering for visualization, as well for inspection of results, as for endoscopic viewings. Notice that if surface-based rendering is parameter-free, it critically relies on the result of the segmentation.

## 7.2 Measurement Tools

The main target of our path and shape extraction framework is to measure pathologies in tube-shaped objects, like aneurysms in brain vessels, and polyps in the colon. We detail in this section the different tools used for quantification of those pathologies, that are characterized by their sections and volumes. Extracting the shapes of our objects, with the *Marching-Cubes* [104], we use a consequence of the Gauss theorem, discretized on the vertices of the triangulation obtained.

### 7.2.1 Gauss Theorem

As classically [9], volume and section measurements are based on Gauss theorem:

**Theorem 7.1 (Gauss)** *Let $\Omega$ be a subset of $\mathbb{R}^d$, let its boundary $\Sigma$ be a closed surface, and $U$ a differentiable vector field, then:*

$$\int_\Omega divU\, dx = \oint_\Sigma U.N\, d\sigma$$

*where $N$ is the outward normal to $\Sigma$.*

A consequence of Gauss theorem is that the volume $\mathcal{V}(\Omega)$ of $\Omega$ can be simplified as an integral over the boundary $\Gamma$

$$\mathcal{V}(\Omega) = \int_\Omega dx = \frac{1}{3} \int_\Omega \mathrm{div}\,(x)\, dx = \frac{1}{3} \oint_\Gamma x.N\, d\sigma. \tag{7.2}$$

### 7.2.2 Volume Measurement

We assume that a 3D tubular structure has been segmented with a level set method and that $\phi(\cdot, t)$ is known at convergence and denoted by $\tilde{\phi}$. We also assume that centered paths have been computed in the tubular structure.

The volume to be measured is defined by the user who chooses a path and specifies two points $p_1$ and $p_2$ on this path. The computed volume is the volume of the interior region of the tubular structure limited by the two plane section $S_1$ and $S_2$ associated to $(p_1, \Pi_1)$ and $(p_2, \Pi_2)$ and defined by $S_i = \Pi_i \cap \tilde{\phi}^{-1}(\mathbb{R}^-)$ $i = 1, 2$. Here is a step-by-step summary of our algorithm, which is illustrated by figure 7.6.

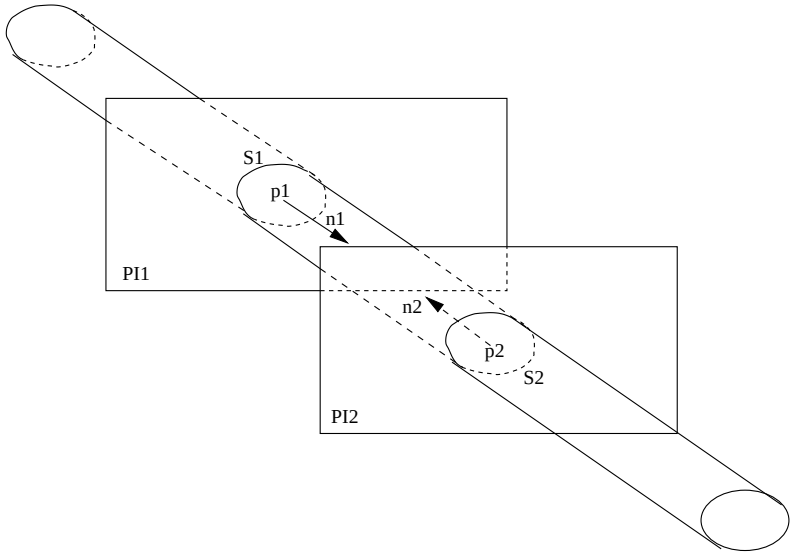



**Figure 7.6. Volume measurement diagram**.

- we compute tangent vectors to the path at $p_1$ and $p_2$, which are the normal vectors $\overrightarrow{n_1}$ and $\overrightarrow{n_2}$ to the plane sections $S_1$ and $S_2$;
- the equations $(p_1, \overrightarrow{n_1})$ and $(p_2, \overrightarrow{n_2})$ of the plane sections $S_1$ and $S_2$ are considered;
- the region of interest is actually the intersection of three subsets of $\mathbb{R}^3$, which are $\tilde{\phi}^{-1}(\mathbb{R}^-)$ and two half-spaces limited by the plane sections;
- we deduce the signed distance functions $\Psi_1$ and $\Psi_2$ to the two half-spaces $\Pi_1$ and $\Pi_2$ from the equations of the plane sections
- Considering that the shape of the object of interest can be complex, and lead to problems of intersection between planes $\Pi_1$ and $\Pi_2$ (see figure 7.8), we define a function $\Psi$ the following way
  1. it is initialized with $\Psi(x, y, z) = \sqrt{3}, \forall (x, y, z)$ in the image domain;
  2. starting from the path point $p_1$ (respectively $p_2$), we apply a region growing algorithm, that labels only the voxels $v$ which have $|\Psi_1(v)| < \sqrt{3}$, (respectively $|\Psi_2(v)| < \sqrt{3}$) and for those voxels, we set $\Psi(v) = \Psi_1(v)$ (respectively $\Psi(v) = \Psi_2(v)$);

3. starting from any path point $p$ not labeled between $p_1$ and $p_2$, we apply a connectivity filter that visits only the voxels where $\tilde{\phi}(v) < \sqrt{3}$ and that are not already visited by the first connectivity filter; and for each voxel visited we set $\Psi(v) = \tilde{\phi}(v)$; it enables to avoid including the interior of undesired parallel structures in the region of interest;

- the region of interest is equal to $\Psi^{-1}(\mathbb{R}^-)$, and a polygonal approximation of its boundary is computed by extracting the zero-level set of $\Psi$;
- the volume of the region of interest is computed using the following decomposition of equation (7.2) on the polygons of the extracted surface:

$$\mathcal{V}(\Psi^{-1}(\mathbb{R}^-)) = \frac{1}{3} \sum_{\mathrm{i}} \mathrm{g_i}.\,\mathrm{N_i}\,\sigma(\mathrm{P_i})$$

where $g_i$, $N_i$ and $\sigma(P_i)$ respectively denote the center of gravity, the outward normal and the surface of the polygon $P_i$.

The overall computation times are very short (less than 3 seconds for a $256 \times 256 \times 60$ image on a SunBlade 100), and the results on basic geometric primitives are excellent in terms of accuracy.

### 7.2.3 Example of volume measurement: an aneurysm

In this case, shown in figure 7.7, where the problem studied is the cerebral aneurysm of figure 6.4, the measurement of the aneurysm volume is done using one trajectory extracted inside a mask defined by the segmentation obtained in figure6.6. Taking two

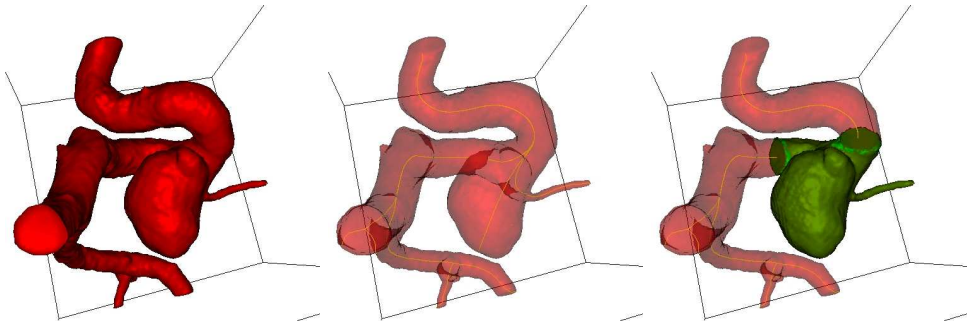


**Figure 7.7. Measuring the volume of an aneurysm**: The dataset used for this segmentation is shown in figure 6.4. Left image is the segmented object obtained in figure 6.6 by combining *Fast-Marching* and *Level-Sets* methods; middle image shows the multiple paths extracted; right image shows a sub-volume of the aneurysm which has been isolated.

positions along the trajectory, we can easily define a volume of interest that contains the aneurysm. The volume shown in figure 7.7-right is not restricted to the aneurysm itself, and contain the surrounding vessel. But a good approximation can be given, by subtracting an approximate vessel volume, using the surfaces of the sections $S_1$ and $S_2$. Advantage of using our connectivity algorithm to obtain $\Psi(\mathbb{R}^-)$ instead of taking

the region delimited by $\tilde{\phi}^{-1}(\mathbb{R}^-)$ and the two half space $\Pi_1$ and $\Pi_2$ determined by the distance functions $\Psi_1$, $\Psi_2$ is illustrated by figure 7.8 on the same dataset.

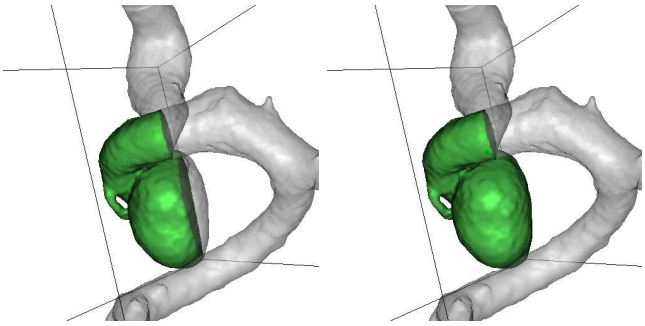


**Figure 7.8. Advantage of the connectivity algorithm**: left image shows the result of computing the intersection of $\tilde{\phi}^{-1}(\mathbb{R}^-)$, $\Psi_1^{-1}(\mathbb{R}^-)$, $\Psi_2^{-1}(\mathbb{R}^-)$. Right image is the representation of $\Psi(\mathbb{R}^-)$ superimposed on the segmentation along the same trajectory, between the same extremities.

### 7.2.4 Section Measurement

We can also apply equation (7.2) in 2D to evaluate the surface limited by a closed planar curve. In order to illustrate this method, we show its application to a phantom dataset.

The data, shown in figure 7.9-left is the acquisition of a cube of Perspex (a type of plastic) with an aluminum rod in it, inside a dead human head. It was acquired

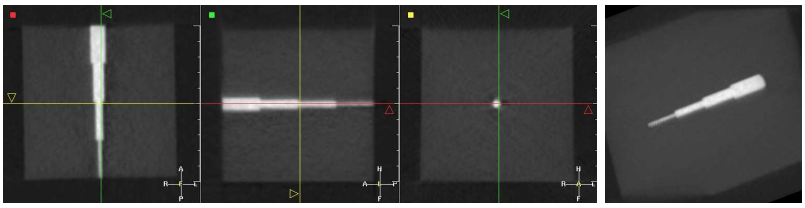


**Figure 7.9. 3D-RA Phantom**: On the left image are shown three orthogonal views of the perspex cube acquired with a **3D-RA** system; right image is a **MIP** view of this data-set.

with the Philips Integris **3D-RA** system. A **MIP** view in figure 7.9-right enables to see the variable section of the aluminum rod.

Following the results of chapter 6, we first segment the phantom with the *Fast-Marching* algorithm, starting from one point at the top of the aluminum rod. Computing the Euclidean path length while propagating, as detailed in section 2.2.3, it is very easy to extract the largest centered path, using the method described in section 2.3, with the thresholded distance $\tilde{\mathcal{D}}$ to the object borders. This path extracted is visible

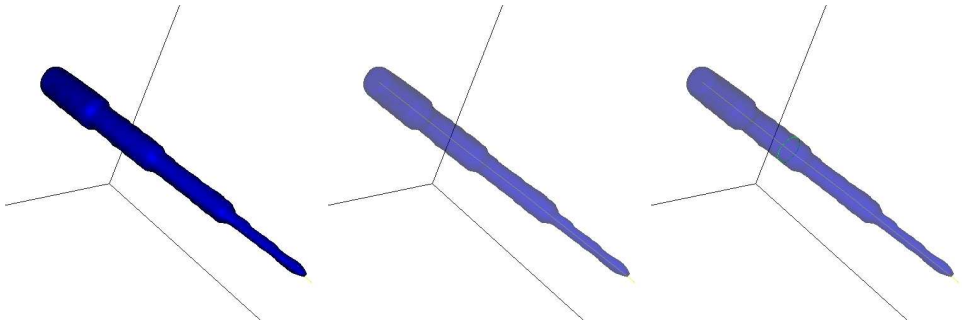


**Figure 7.10. Segmentation result on the 3D-RA Phantom of figure 7.9**: Left image is the segmented object with the combination of *Fast-Marching* and *Level-Sets* methods; middle image is the same object with opacity $< 1.$ and the path extracted; right image shows the intersection of the phantom surface with the section plan for measurements.

in figure 7.10-middle, by transparency. In a few iterations, the *Level-Sets* algorithm, with region-based forces, gives the result shown in figure 7.10-left.

In the experimental tool we built, the user specifies a particular path and obtains the section of the tubular structure according to the length of the path. The path is supposed to have a discrete representation, i.e. is represented by a list of points. Here we give a summary of the performed calculations for each point of the path:

- the normal of the section plane is computed using an approximation of the tangent vector to the path;
- an orthonormal base of the section plane is deduced;
- a rectilinear 2D grid, centered on the current path point, is defined on the section plane;
- at the center of each cell of the grid, the value of $\tilde{\phi}$ is computed by interpolation;
- an adequate algorithm is used (we used the Marching Squares) to compute an approximation of the zero iso-contour in the 2D grid;
- the surface enclosed in the resulting polygonal line, which in our example is drawn on the surface in figure 7.10-right, is computed thanks to a decomposition of equation (7.2) on the polygonal line.

Like in the case of volume measurements, the computation times are very short, and the algorithm gives very accurate measurements of basic geometric primitives. Concerning the phantom problem, we have computed this section at each path point (see figure 7.11). Figure 7.11 shows the measures done along the path displayed in figure 7.10-right. On the graphic, we have displayed the several real dimensions of the aluminum cylinders, and we have also displayed the interval of deviation of 2% that was indicated by a study on the accuracy of the calibration, the distortion correction,

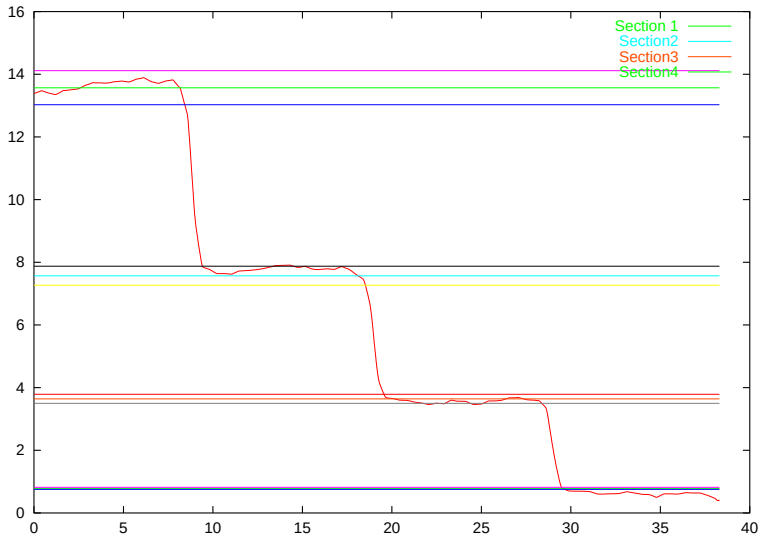



**Figure 7.11. Section measurements along the segmented phantom of figure 7.10**: It represents the values of the section across the trajectory extracted, with the deviation of 2% superimposed.

and the reconstruction of the **3D-RA** system [83]. The section measurements of the segmented object show that our method gives results which lie in those intervals, when the radius is more than one millimeter.

# Chapter 8

# Tree Extraction framework

**Résumé** — Dans la partie précédente nous avons présenté un algorithme mettant en oeuvre une collaboration entre le *Fast-Marching* et les *Level-Sets* pour la segmentation. Dans ce chapitre, nous souhaitons présenter une application de cette collaboration spécifiquement dédiée aux structures arborescentes du type arbre vasculaire.
Tout d'abord nous montrons comment le *Fast-Marching* permet de fournir une présegmentation rapide et précise pour les structures arborescentes dans la section 8.3.
Nous utilisons ensuite les *Level-Sets* de la même manière que dans la section 5.4 de la partie précédente.
Finalement nous montrons comment le *Fast-Marching*, déjà utilisé pour l'extraction de trajectoires dans la partie I, permet aussi d'extraire plusieurs trajectoires et de remonter à l'information d'arbre ou de squelette d'un objet tubulaire avec embranchements multiples.

**Abstract** — In the previous part I, we detailed an algorithm using *Fast-Marching* and *Level-Sets* in a collaborative manner for object segmentation. In this chapter, we introduce an application of this collaboration specifically adapted to tree anatomical structures, like vascular or arterial tree. First of all, we demonstrate in section 8.3 the ability to build a fast and accurate pre-segmentation for those tree structures using a dedicated *Fast-Marching* algorithm.
We further apply the *Level-Sets*, as in section 5.4, for converging to a more accurate solution.
Finally, we show in section 8.4 how the *Fast-Marching* ability to extract trajectories, as used in part I, can be extended to the simultaneous extraction of multiple trajectories, and to obtain the underlying tree structure of a tubular anatomical shape with several branches.

## 8.1 Introduction

In the first part of the thesis, we have implemented several techniques to extract a trajectory inside a tubular structure. We have shown application of this fast minimal path-extraction process to automatic and interactive methods to extract lineic structures in images. In the second part of the thesis, we have combined fast and accurate methods for shape extraction, using the same kind of grey-weighted distance transform algorithms. We have proved the ability of those techniques to extract surfaces, and to emphasize pathologies, in several applications. In the last part of the thesis, we now want to integrate the path and surface extraction algorithms, in order to present an accurate global framework for the segmentation, the visualization, and the quantification, of anatomical objects. In the previous chapter, we have detailed the algorithmic techniques to obtain representations and measures of our anatomical objects, based on extracted primitives of our objects like shapes and skeletons. In this chapter, we will present the basic framework, and extend its possibilities to the detection of tree-like structures, and their corresponding set of multiple trajectories, in order to enhance measures and visualization of pathologies of any tube-shaped object.

This chapter will be illustrated by applications of the algorithms presented on the segmentation and quantification of vessels in contrast-enhanced 3D medical images.

## 8.2 Motivation

We have seen in part II a method to use front competition for image segmentation. This process involved to visit the whole image domain, and was not tuned for a particular category of objects. Moreover, in huge images, as multi-slice CT scanners (see application to lungs images in chapter 9.2), the visit of the whole image cannot be done in interactive time.

### 8.2.1 Tree extraction

In this chapter, we are focusing on the extraction of thin tubular structures. Our algorithms can be dedicated to this particular category of tube-shaped objects. If the propagation of a front could be restricted to the part of the image occupied by those structures, the computing time could be divided by almost 5, since vessels in a typical MR-Angiography image do not exceed 10% of the whole volume.

In chapter 2, we have developed an algorithm that can be the basis of this kind of tubular shape extraction object: a technique to evolve a front inside an object of interest and compute at the same time the Euclidean distance to the start point. It was used to reduce the user interaction to locating only one extremity of the path inside a tubular structure. This Euclidean distance can be used to stop the front propagation inside the desired object. If we have precise knowledge of its length, we can decide to stop when this given length has been reached in the expression of the Euclidean path length computation, as explained in section 2.2.3. The result of this technique is shown in figure 8.1.

**Figure 8.1. Segmenting the colon volume with simple front propagation**: as in the virtual endoscopy facility, the user locate a starting point at one particular recognizable part of the colon, then a front is propagated from this seed point until a maximum path length is reached. Left image represents the datasets where the intersection with the segmented object is visible in pink. Right image is the 3D volume rendering of the final segmentation.

However, classical segmentation problems do not provide an excellent contrast like the air-filled colon on a CT scanner, and the propagation cannot stick to the object walls, as it is shown in figure 3.6. For example, if we apply the same kind of propagation in the dataset shown in figure 3.12 for the endoscopy application in chapter 3, the corresponding wave propagation looks like figure 8.2. The front floods

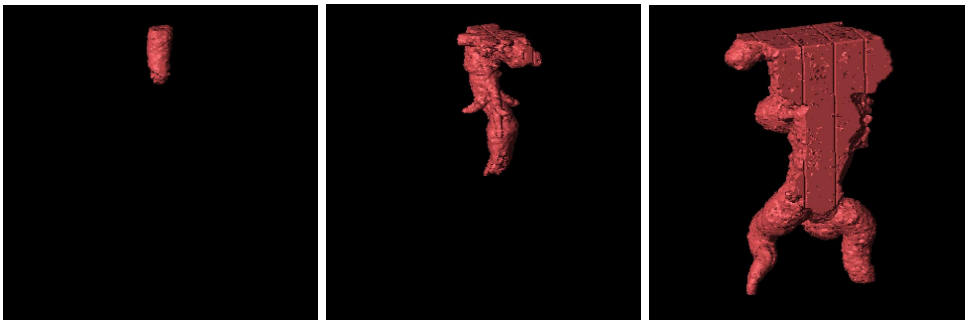


**Figure 8.2. Wave propagation inside the aorta MR dataset**: These three images represent different steps of the propagation inside the aorta MR dataset using *Eikonal equation* equation with a potential similar to the one defined for the endoscopy application (a simple function of the grey levels either linear or non-linear).

outside the object and cannot be used as an initialization step for a more complex segmentation, like the combination of the *Fast-Marching* and the *Level-Sets* which was presented in the previous part.

In the following section, we will present a new algorithm, based on the *Fast-Marching* and dedicated to a *quick and dirty* segmentation of the tree structures in 3D medical images.

## 8.3 Design of an adequate initialization algorithm

We have shown the possibility to provide efficiently an initialization for more complicated methods in the previous part of the thesis. Setting up a framework for the visualization and the quantification of thin tubular structures, based on the same combination of the *Fast-Marching* and the *Level-Sets*, we show in this section how the previous initialization step, which is not tuned for this kind of thin and long objects, can be specifically optimized for this target.

### 8.3.1 Propagation Freezing for Thin Structures

*Freezing* a voxel during front propagation is to consider that it has reached the boundary of the structure. When the front propagates in a thin structure, there is only a small part of the front, which we could call the "head" of the front, that really moves. Most of the front is located close to the boundary of the structure and moves very slowly. For example voxels that are close to the starting point, the "tail" of the front, are moving very slowly. However, since the structure may be very long, in order for the "head" voxels to reach the end of the structure, the "tail" voxels may flow out of the boundary since their speed is always positive. This is illustrated in the example of figure 3.12. If we apply fast marching in the dataset shown in figure 3.12-top with a potential based on the gray level with contrast enhancement the corresponding wave propagation looks like figure 8.2. The front floods outside the object and does not give a good segmentation.

For these reasons, it is of no use to make some voxels participate in the computation of the arrival time in *Eikonal equation* by setting their speed to zero, which we call *Freezing*. First step is to design the appropriate criterion for selecting voxels of the front which needs *Freezing*.

Concerning the application to the tree tracking, the several improvements brought by this method are

- to accelerate the computations, by visiting a very small number of voxels during propagation;
- to enable the segmentation of thin tubular structures;
- therefore enabling the centering inside those tubular structures.

First step is do design the appropriate criterion for *Freezing* voxels of the front. We illustrate this *Freezing* principle on a synthetic branching structure in 2D.

#### Synthetic test problem

A synthetic example of a tree structure is shown in figure 8.3. In this case, setting an initial seed point at the hierarchy, we would like to extract in a very fast process the multiple branches of the structures, and its corresponding skeletonization, in a single process. Figure 8.3 shows the result of the classical front propagation technique with the *Fast-Marching* coupled with a maximum Euclidean path length stopping criterion. The action map displayed clearly indicates that the domain visited is a whole

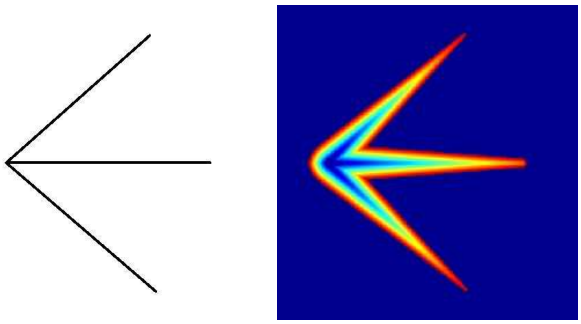


**Figure 8.3. Synthetic test problem**: The left image is the medium where a front has been propagated, starting at the root of the three branches, and stopping when a maximum distance criterion of 300, computed according to method described in section 2.2.3, has been reached; Right image is the corresponding action map.

"blob-like" structure where the underlying tubular shape is somehow lost. Therefore, tracking a minimal path from the regional maxima of the action map will not lead for sure to paths that stay inside the object of interest. It emphasizes the little use of this method, without a clear constraint on the domain of points visited.

### Using Time for Freezing

The heuristic presented in this section is to discriminate the points of the front that are spending a long time in the propagating front, i.e. points that are visited but whose action is not *frozen*, in the sense defined in table 2.1.

Unfortunately, this criterion is very difficult to manage, as shown in figure 8.4. The results are non-predictable, and this is probably because the time spent in the

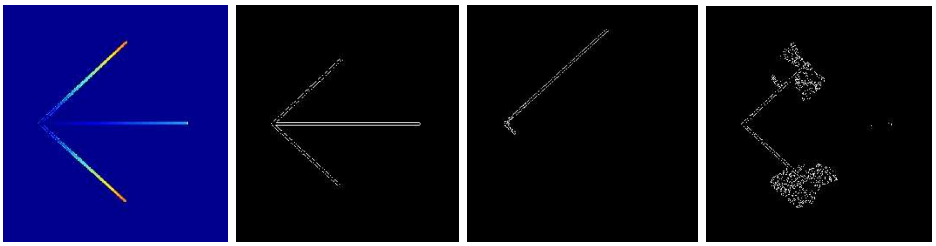


**Figure 8.4. Instability of the Time criterion for Freezing**: Left image is the action map obtained with a maximum time criterion of 100 iterations; Other images are freezing maps (white pixels) with respectively from left to right 100, 80 and 60 iterations as maximum time spent in the front.

front for a voxel is related to the local cost of the propagation at this voxel, but do not have any relation with the position of the voxel relatively to the object that we are trying to segment.

### Using Distance for Freezing

The distance to the start point is a direct output from the method we already developed for reducing user-intervention in the *Virtual Endoscopy* process in sections 2.2.3 and 3.1. It seems far more "natural" to use the distance to the starting point, or relatively to the most far propagating part of the front, since this notion is completely embedded in the topology of the object we are trying to extract: the section of a tube-shaped objects must be small towards its extent. We must discriminate the points of the front that are near the initializing seed points while other parts of the front are already far. It will prevent from flooding in non-desired area of the data.

We can fix several criterion for the *Freezing* based on the distance. Knowing the current maximum Euclidean path length $d_{max}$ in the front propagation process we can decide that a voxel $\mathbf{v}$ of the propagating front (i.e. *Trial* voxels) should be removed from the front (i.e. set as *Alive* voxel):

- if $\mathcal{D}(\mathbf{v}) < d_{max}/\alpha$, with $\alpha \geq 1$ user-defined; or
- if $\mathcal{D}(\mathbf{v}) < \max(d_{max} - \tilde{d}, 0)$, with $\tilde{d} > 0$ chosen.

The results are now predictable, in the sense that the Euclidean distance to the starting point is a measure which contains information about the geometry of the surface extracted, and in particular its length. This is less related to the local cost of the propagation in each voxel, and more to the position of this voxel in the object. This distance criterion has proven reliability as well in 2D as in 3D, and we worked upon its implementation in the following. A 2D example on the synthetic test is shown in figure 8.5.

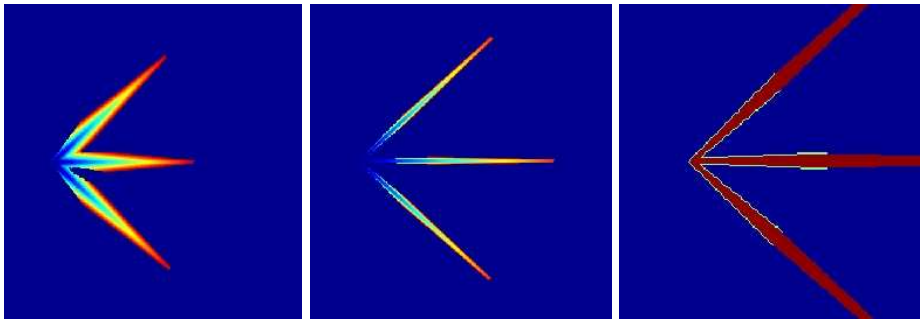


**Figure 8.5. Using Distance for Freezing**: Left and middle images are action maps with distance criterion of respectively 100 and 50; right image is a zoom on the freezing map for a distance criterion of 50: the pruned points are set in green.

### Algorithmic implementation of the *Freezing*

Once the criterion has been chosen, at each time step we insert our visited points both in the classical action related heap, and in another data-structure where the ordering key is the criterion. As for the action, we can use a min-heap data-structure, since the partial ordering provided is sufficient.

At each iteration, we are able to remove all the points whose keys are greater/lower than this criterion, starting from the minimum/maximum element in the tree. It can be implemented easily for the time criterion by recording the iteration at which any point has been inserted in the heap, and to store this time in another min-heap data-structure. Therefore, the element at the top of the heap will still be the point that has spent the longest time without being evolved to the *Alive* set. For the distance criterion, the min-heap key is the computed distance, which means that the element at the top of the heap will still be the point that is the nearer *Trial* point to the starting point.

In the following is detailed an algorithmic implementation of the *Freezing* with the second criterion for the distance information.

**Definition**

- a starting point $\mathbf{p}_0$, located at the root of the tree structure;
- the usual set of data-structures for front propagation, including an action map $\mathcal{A}$, one min-heap structure $\mathcal{H}_\mathcal{A}$ and a penalty image $\mathcal{P}$ which will drive the front propagation, and which is a function of the position only;
- a distance map $\mathcal{D}$ to compute the Euclidean minimal path length, as explained in section 2.2.3;
- another min-heap data structure $\mathcal{H}_\mathcal{D}$, where the ordering key for any point $\mathbf{p}$ is the value of $\mathcal{D}(\mathbf{p})$, which means that the first element of this heap will be the *Trial* point with smallest distance $\mathcal{D}$;
- several counters $d_{max}$, $\tilde{d}$, $d_{stop}$

**Initialization**

- initialize the classical front propagation method, setting $\mathcal{A}(\mathbf{p}_0) = \mathcal{D}(\mathbf{p}_0) = 0$ and storing the seed point $\mathbf{p}_0$ in both min-heap structures $\mathcal{H}_\mathcal{A}$ and $\mathcal{H}_\mathcal{D}$;
- $d_{max} = 0$
- $\tilde{d}$ and $d_{stop}$ are parameters for tuning the algorithm (user defined).

**Loop:** at any iteration

- Let $\mathbf{p}_{min}$ be the *Trial* point with the smallest action $\mathcal{A}$;
- proceed according to the classical *Fast-Marching* algorithm, by examining its neighbors, and updating the min-heap $\mathcal{H}_\mathcal{A}$ with the new action values computed;
- take $d_{max} = \max\left(d_{max}, \mathcal{D}(\mathbf{p}_{min})\right)$;
- consider $\mathbf{q}_{min}$, the first element of $\mathcal{H}_\mathcal{D}$, being the *Trial* point with the smallest distance $\mathcal{D}$. While $\mathcal{D}(\mathbf{q}_{min}) < \max\left(d_{max} - \tilde{d}, 0\right)$ do
  - set $\mathcal{D}(\mathbf{q}_{min}) = \mathcal{A}(\mathbf{q}_{min}) = \infty$;
  - set $\mathbf{q}_{min}$ in the *Alive* set, then $\mathbf{q}_{min}$ will not be used for computing the action/distance at its neighbors location.
  - delete $\mathbf{q}_{min}$ in both $\mathcal{H}_\mathcal{D}$ and $\mathcal{H}_\mathcal{A}$;
- if $d_{max} > d_{stop}$, exit the loop.

This heuristic is to discriminate the parts of the front that are propagating slowly, by recording the maximum distance which has been traveled, and compare it to the distance which has been traveled by this parts. If the ratio between those two distances is two important ($>$ given threshold), we "freeze" those parts by setting there speed artificially to zero. It enables to stay inside the object when it is long and thin like tubular structure, as shown in figure 8.5. The domain visited by our algorithm is slightly smaller than the previous one (figure 8.4-right) and this domain shortens with the distance criterion, when we compare left and middle images in figure 8.5. The figure 8.5-right clearly demonstrates than the *Freezing* principle discriminates the points located far from the propagating fronts.

**Illustration on the Vascular tree extraction problem**

The method explained previously is very useful when it is used for vascular segmentation. Initialization step is therefore performed in a very fast manner by just setting a seed point at the top of the tree hierarchy. Figure 8.6 displays results of this algorithm. The distance threshold is a parameter which is not very sensitive: we generally

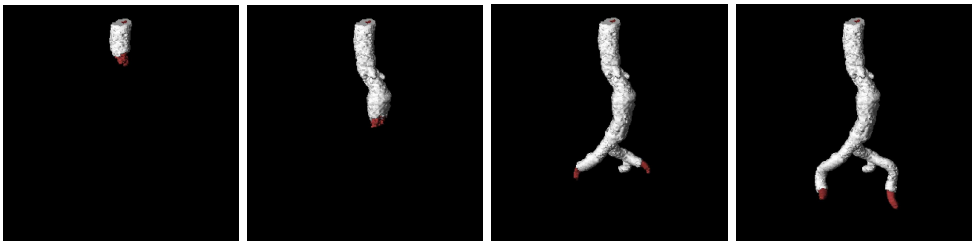


**Figure 8.6. Using Distance for Freezing in the Aorta**: From left-to-right, images show iterations of the segmentation process; the propagating front is in red, and the frozen voxels are in white.

take a value related to the *a priori* dimensions of the object. This threshold must be more important than the assumed maximum section of the object. It will approximately represent the volume of points bounded by connected envelope of the front voxels that are not frozen.

### 8.3.2 Suitable stopping criterion

Having designed an adequate criterion for *Freezing* the unwanted parts of the front that could lead to "flooding" of the evolving wave in other parts of the image, we now explain our strategy to stop automatically the propagation.

Previous strategy was to use a maximum Euclidean path length to stop propagation, like for the virtual endoscopy application. In *Virtual Endoscopy* , the user can set both extremities of the trajectory, if he has an *a priori* knowledge of the anatomical objects. Extraction of tree-like structures cannot use such an assumption: the number of branches in our structure is undefined, only assumption being that the user can fix a point inside the structure, at the beginning of the segmentation process.

The *Freezing* process will provide a criterion which is independent of the number of different branches to recover. If we plot the maximum distance $d_{max}$ of section 8.3.1, as a function of iterations while propagating, we will observe the following profile shown in figure 8.7. We clearly see that this distance increases linearly until a big decrease of the slope appears. It is important to notice that this shock indicates when

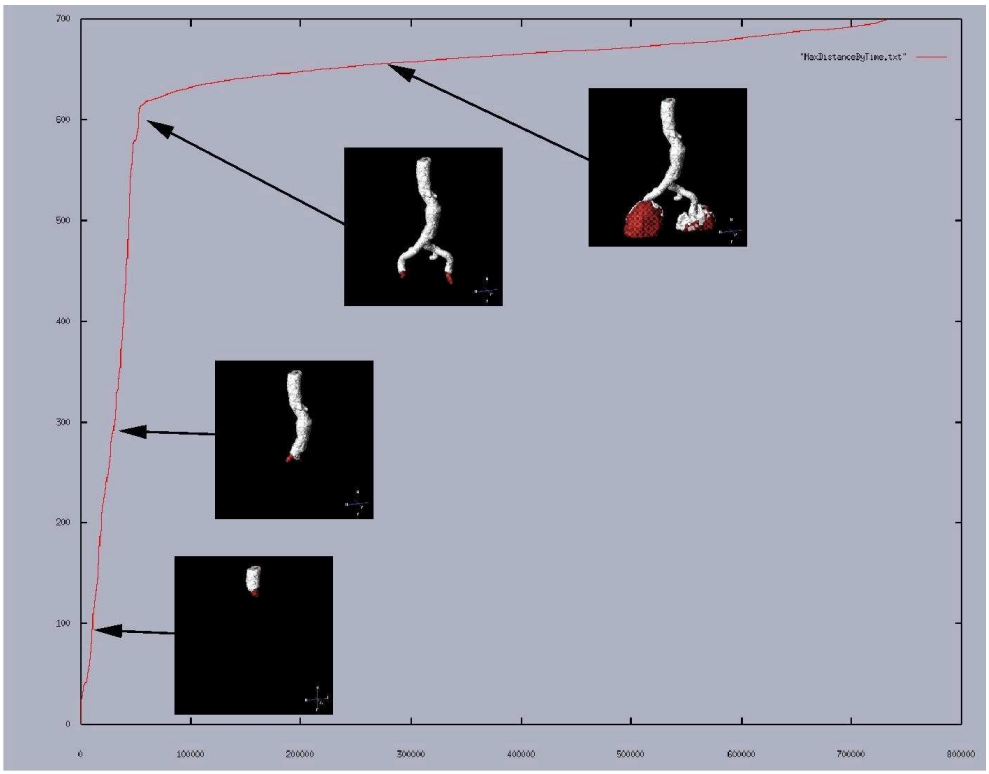



**Figure 8.7. Using Distance for Stopping propagation in the Aorta**: The images of the propagating fronts of figure 8.6 are super-imposed on the evolution of the maximum distance crossed by the front propagation across iterations; it emphasizes that the decrease in the slope is related to the "flooding" out the aorta.

the front flows out of the object at "heads" of the front. We decide to stop front propagation at this particular time. During the first part of the plot, the function is quasi-linear. The slope is directly related to the section area of the tubular object. By definition of Fast Marching, the number of iterations is equal to the number of voxels that are alive. It means that passing through a certain length in the aorta implies to visit a number of voxels proportional to the length.

Let us assume that the global section of our aorta is constant in our dataset. This is approximately true in large parts, but becomes a wrong assumption in the very thin parts of the vessels and arteries. But we can assume that the front propagates at the same speed inside the object. Therefore, the number of voxels visited is proportional

to the section area. Then the slope collapse can be easily detected using a simple threshold on the slope, depending on the object we want to extract. Even if there are aneurysms in the data set, and even if the mean section of the object increases with the depth, we can assume that we do not want to extract an object which is twice the maximum section. We could then derive a criterion on the maximum section of the object $S_{max}$ which is obviously related to the section area of the object of interest. Recording the first iteration where the front flows out, it gives us the maximum distance where we must stop propagation.

## 8.4 Extracting the skeletal information

In the following, we assume that we use the *Fast-Marching* and the *Level-Sets* in a collaborative manner, in five steps:

1. the user input is a seed point for region-growing;

2. the *Fast-Marching* using the *Freezing* principle is evolved from this starting point;

3. this evolution is stopped using either the distance, the user intervention, or an automatic criterion;

4. the binary mask defined by the propagation gives the initialization of the region based descriptors $k_{in}$ and $k_{out}$, as used in section 6.2;

5. the *Level-Sets* model is evolved with equation 4.6 for a small number of iterations.

The process is really similar to the framework detailed in section 5.4. The *Fast-Marching* using the *Freezing* principle will act as a rough initialization step, which will provide the binary image of the voxels visited. This mask will also serve to initialize the different probabilities of the region descriptors defined in section 5.2. First row in figure 8.8 shows the surfaces of several tubular structures extracted with the *Freezing* algorithm. The domain of voxels visited during this first step is used to set correctly the descriptors of the *Level-Sets* model, that converges in a few iterations to the surfaces which are shown in the second row in figure 8.8. Notice that the scheme used here is in equation (7.1), where forces have been included to restore the distance function.

In this chapter, we do not implement dedicated algorithms based on the *Level-Sets* methods. They are used in a very classical manner, to converge to sub-pixel accuracy results, on the basis of as-hoc fast algorithms. However, the level of accuracy that is achieved by the *Level-Sets* cannot be of course outperformed by the initialization method. The convergence step they achieved cannot be replaced in any way by the *Fast-Marching*.

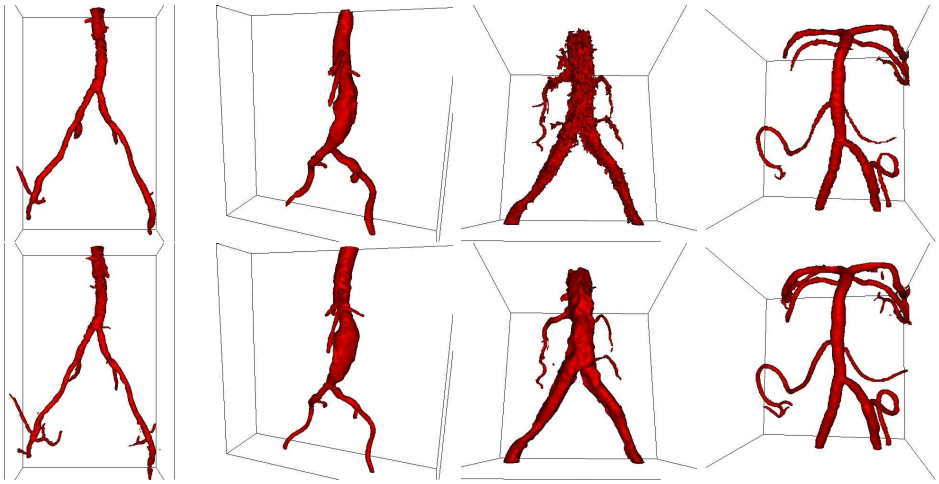


**Figure 8.8. Final segmentation of vascular objects**: First row shows different vascular objects that have been extracted with the *Freezing* algorithm - except the example shown in last column of the right, where the method used was the competitive fronts algorithm; Second rows is the final result of the segmentation after 40 iterations.

### 8.4.1 Combining path and shape extraction

The complete framework for path and surface extraction we have developed will be illustrated in this section by results on a **3D-RA** acquisition of a stenosed vessel, which is shown in figure 8.9.

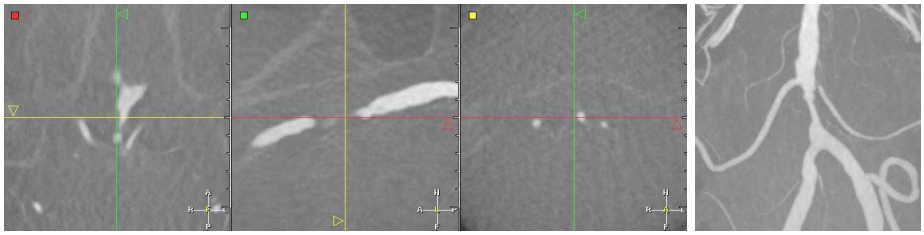


**Figure 8.9. 3D-RA dataset of an aortic stenosis**: left image shows three orthogonal views of the dataset; right image is a **MIP** view of the same dataset.

We have shown in the part I of the thesis how to extract a trajectory inside a tubular structure. In part II of the thesis, we have combined fast and accurate methods for shape extraction of tube like objects. We now want to combine the results of both parts, and extend this facility to the detection of tree-like structures, and their corresponding set of multiple trajectories, in order to enhance measures and visualization of pathologies of any tube-shaped object.

We worked upon the extension of the trajectory extraction method, applied for example to virtual endoscopy in chapter 3 to the case of multiple trajectories. For

example, the dataset in figure 8.9-left is a structure with branches, which pathology - a stenosis - is clearly visible on the **MIP** view in figure 8.9-right. The complete study of this pathology, with minimum interactivity, would be to extract its surface, and all needed trajectories inside it, in order to give accurate measurements.

### Techniques found in the literature

The combination of path and shape representation is a framework already studied as well in Computer Graphics as in Computer Vision. In Computer Graphics, cylindrical shapes description is done by implicit surfaces (in the sense of [13, page 223]) defined by the convolution of a filter kernel with a skeleton. In other words, this *distance surface* is a surface that is defined by distance to some set of skeletal elements, like any curve. But in graphics, the target is to improve visualization and interactivity over the representation of the object. However, it connects to vision because it is often convenient to model a shape as a generalized cylinder as done in [132], for reconstruction of anatomical shapes, as done in [175] by combining the fitting of a generalized cylinder, and its symmetry axis.

In those methods, the central axis constrain the extraction, and models the tubeness of the final object extracted.

### Our multiple path extraction method

In our case, the shape is initialized by *Fast-Marching*, thus a path construction method, but we are going to use the solution at convergence of the *Level-Sets* in order compute the final set of trajectories - i.e. the skeletal information of our object. Therefore, shapes controls path extraction. This is exactly the kind of methods that lead to accurate measurements and visualization of the objects:

1. It relies on a sub-pixel shape extraction model; thus the intersection of a cross section plan and the surface is an improved measure of the objects, while cylinders approximate the model.
2. The *Level-Sets* enables any change in topology, and there is no constrain on the initialization of the model, how huge can be the number of branches in the anatomical object.
3. The paths used for quantification are based on this robust surface extraction model, increasing the robustness of the measures.
4. The user input is limited to the setting of the root of the tree hierarchy.

Our method is based on the construction of a connectivity map, by looking at several chosen iterations to the connectivity of the propagating front (i.e. the *Alive* voxels) and the connectivity of the sets of voxels visited, as shown in figure 8.10. Defining a distance step, each time this step has been accumulated by the front, we label the different sets of visited voxels, and we thus detect when a front separate, at a branch, into several not connected sets.

When the whole domain has been visited, we take for each separate set a representative voxels, which is the most far from the starting point, and we set it as

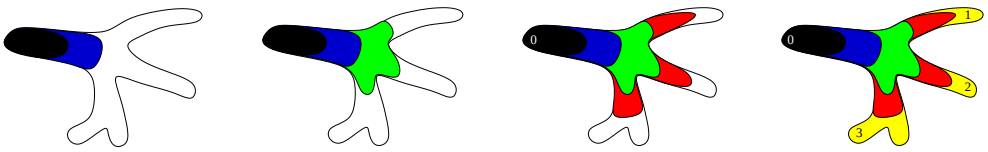



**Figure 8.10. Multiple path extraction algorithm**: From left to right, these are connectivity tests made on the propagation of a wave inside a segmented object, starting from the voxel designated by label 0, until the whole domain is visited.

an applicant extremity for back-tracking a trajectory. Notice that the distance step defines the accuracy of the method, since a too important step will lead to misunderstandings: on the right image, only the extremity designated by the label 3 will be eligible for back-tracking, while there are two branches, because the distance step is bigger than both branches.

### Multiple Path Extraction Algorithm

The algorithm we devised for multiple path extraction is mostly inspired from works on skeletal extraction from binary, or scattered data. It can be easily compared to morphological processes, but has two advantages: we can choose the *scale* or accuracy of the multiple path extraction, and we can derive this *scale* from anatomical knowledge of the data studied. It is a complete framework in the sense that, the path extraction relies here on a segmentation process which can be as well handled by the *Fast-Marching* or the *Level-Sets* methods. This segmentation step defines a binary mask $\mathcal{M}$ which is one of the main input of our algorithm:

**Definition**

- a binary mask $\mathcal{M}$ which defines the region of interest in the image;
- a penalty image $\mathcal{P}$ which will drive the front propagation;
- a distance map $\mathcal{D}$, computed with the method described in section 2.2.3, and a distance step d, user-defined parameter that controls the accuracy of the end-point extraction;
- a counter $c_d$ that recalls the iteration number of the loop in our algorithm;
- a label map $\mathcal{L}$ to label each branch detected, $n_{\mathcal{L}}$ a label counter, and an array $\mathcal{E}$ which will recall the hierarchy of the branches detected;
- a starting point $p_0$, located at the root of the tree hierarchy.

**Initialization**

- $\mathcal{M}(i,j,k) = 1$ for all voxel in the region of interest, elsewhere $\mathcal{M}(i,j,k) = 0$;
- $\mathcal{L}(i,j,k) = -1$ for all voxel in the image domain, $n_{\mathcal{L}} = 0$, and all elements of $\mathcal{E}[i]$ are set to $-1$;
- We initialize the usual set of data-structures for front propagation, including an action map $\mathcal{A}$, the distance map $\mathcal{D}$, and a min-heap structure;
- we initialize a classical front propagation method, setting $\mathcal{A}(p_0) = 0$ and storing $p_0$ in the min-heap structure;  item the counter $c_d = d$.

**Loop**

- we propagate the front with Eikonal equation, computed with penalty $\mathcal{P}$ on the domain defined by the mask $\mathcal{M}$;
- for each *Trial* point $p$ visited in the *Fast-Marching* algorithm, $\mathcal{L}(p)$ is set to the label of its current *Alive* neighbor with minimal action;
- if we visit a voxel $p$ with $\mathcal{D}(p) \geq c_d$:
  1. we consider the set of *Trial* points $\mathcal{T}$, that are all stored in the min-heap data structure, we consider $t_1, \ldots, t_k$ its k subsets of connected components (with 26-connexity in 3D), obtained through a simple connectivity algorithm;
  2. In all subset $t_i, i \in [1, \ldots, k]$
     - considering the old label $l_{i_{old}}$, and the new label $l_{i_{new}}$, we set $n_{\mathcal{L}} = n_{\mathcal{L}} + 1$, $l_{i_{new}} = n_{\mathcal{L}}$, and $\mathcal{E}[l_{i_{new}}] = l_{i_{old}}$;
     - for all the points $p \in t_i$, we set $\mathcal{L}(p) = l_{i_{new}}$;
  3. $c_d = 2 \times c_d$;
  4. we stop if the whole domain defined by $\mathcal{M}$ is visited.

**Termination**

- we consider all sets $\mathcal{L}_j$, $j \in [1, \ldots, n_{\mathcal{L}}]$ defined by the label map $\mathcal{L}$ with different labels $l_j$;
- we select the subset of $\mathcal{L}_k$, $k \in [1, \ldots, n_{\mathcal{L}}]$, which have $\mathcal{E}[l_j] \neq -1$ and $\forall n \in [l_j; n_{\mathcal{L}}]$ $\mathcal{E}[n] \neq l_j$;
- $\forall \mathcal{L}_k$ selected, we find the voxel $(i, j, k)$ with maximum distance $\mathcal{D}(i, j, k)$ and set it as end point for back propagation;
- we back-propagate from all final voxel selected and extract a set of multiple trajectories.

Figure 8.11 shows several label maps $\mathcal{L}$ with $c_d = 10$, 30 and 50. $c_d$ is the minimum

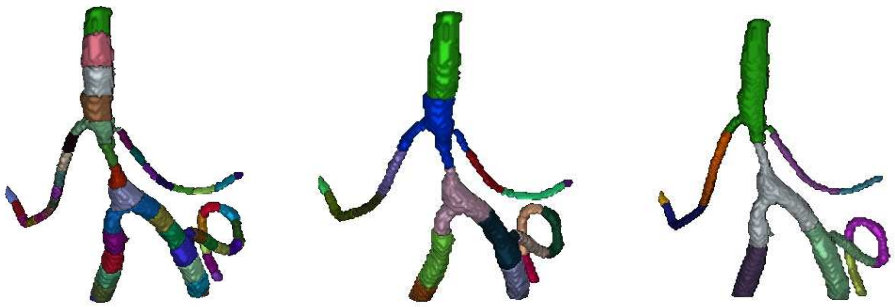


**Figure 8.11. Labeling algorithm for multiple path extraction**: From left to right, the images show the label map obtained with the multiple path extraction routine applied with path steps 10, 30 and 50 respectively.

size of the branches detected, it is the *scale* of the algorithm accuracy. If this scale is chosen small, lots of branches will be detected, but if the scale is increased, the computation time will decrease as well, because it controls the number of connectivity tests which are performed on the *Trial* voxels, during propagation.

**illustration on the vascular tree extraction**

In figure 8.12, one can observe the complete framework of *Fast-Marching* initialization followed by several iterations, using *Level-Sets* methods, and finally, the extraction of multiple trajectories inside two different datasets. The computations for the paths are

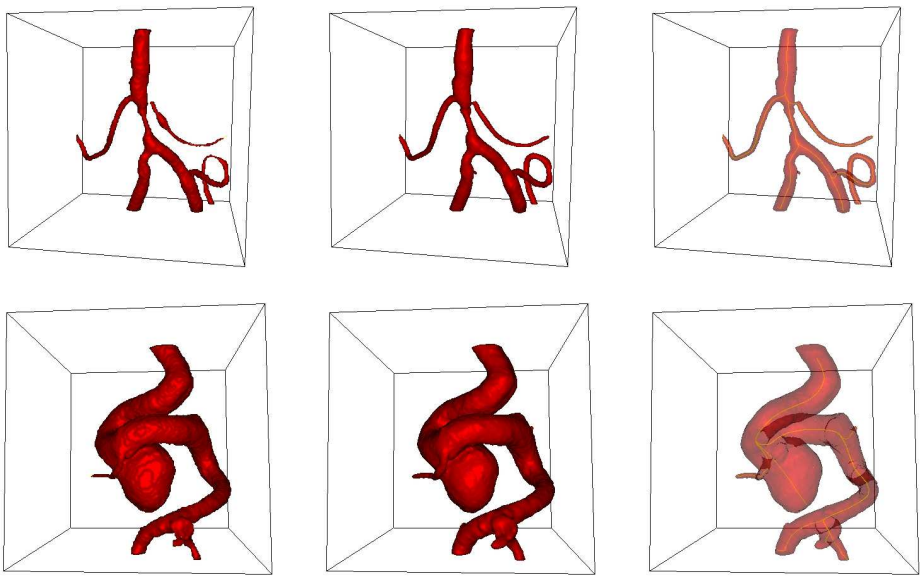


**Figure 8.12. Complete method applied to several objects**: First row is the framework applied to the stenosed object of figure 8.9 and second row concerns the aneurysm shown in figure 6.4 - Left column is the initialization given by the *Fast-Marching* method; middle column is the surface obtained after a small number of iterations of the *Level-Sets* method; right column shows the multiple trajectories extracted with the labeling algorithm, by transparency.

restricted to a small number of points, located inside the objects of interest (usually less than 20% of the whole volume, leading to interactive computing times. Those paths are already very useful for virtual inspection of pathologies, for example in the aorta (as done in the section 3.1), or measurements along the trajectories extracted, using the techniques detailed in section 7.2. Figure 8.13 shows the result of applying the multiple path extraction algorithm explained previously. This set of paths is the basis of the quantification techniques that can be applied on such a dataset (this aorta presents an Abdominal Aortic Aneurysm).

Originality of this algorithm, towards front propagation techniques applied for multiple path extraction, as in [101], where the set of endpoints is manually drawn in the original image. In our case, all trajectories are extracted automatically.

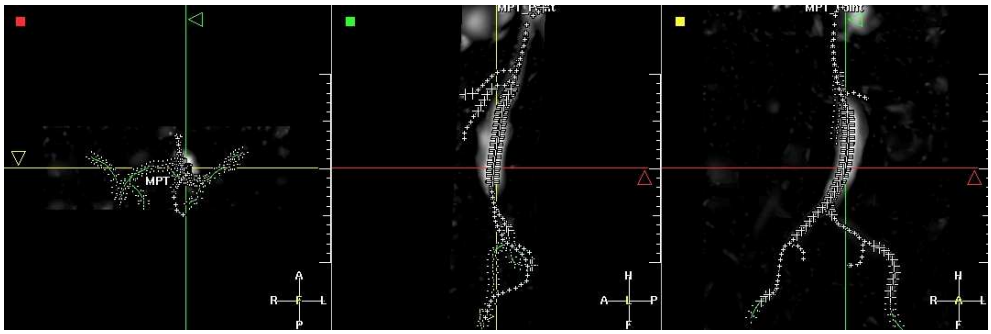



**Figure 8.13. Multiple trajectory extraction from only one seed point**: This figure represents the projection on three orthogonal views of the complete set of trajectories tracked in the aorta MR dataset which was segmented in figure 8.8 in the second row; the *Freezing* method for initialization with the *Fast-Marching* algorithm has been used to extract centered trajectories.

### 8.4.2 From Trajectories to Tree Extraction

The trajectories obtained with our algorithm can guide virtual endoscopes. They can also be used for quantification of pathologies, by measuring the variation of the section of the object, across the curvilinear abscissae of the path extracted. But the information of trajectory is not related to the whole branching structure and is just the minimal centered path between two extremities. Therefore, the user is assumed to know the position in the object of this trajectory. And those trajectories are not related to each other, leading to possible misunderstanding in this position. Moreover, this absence of spatial relationship between the paths and the surface disable the use of further developments like automatic labeling of the branches, and accurate localization of pathologies. In order to extract the information which is relevant in order to analyze the surface of the tree-shaped object extracted, we need to extract the underlying skeleton on the basis of our trajectories, as done in figure 8.14. The process

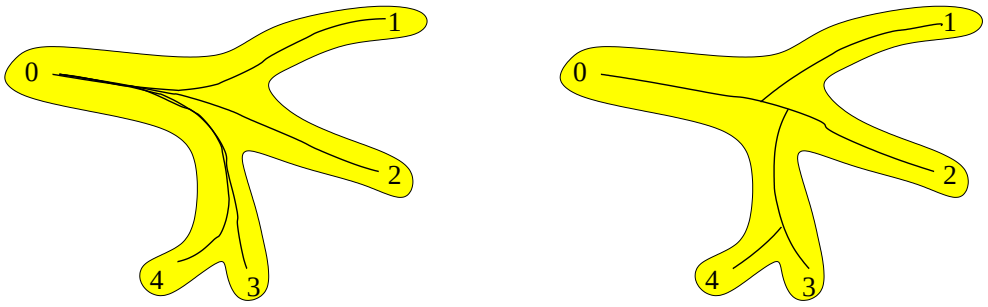



**Figure 8.14. From trajectories to tree representation**: Left image is a set of trajectories extracted in a segmented object. right image represents the valuable tree structure needed for quantification.

of extracting the tree structure from the trajectories is simple: during backtracking of the trajectories, we adjoin those which are close to each other, creating a branching point. The only parameter is the definition of proximity between trajectories.

### Algorithmic implementation

As a second process, we can extract a skeleton of our object, from this set of multiple trajectories. The initialization use the same input than the multiple path extraction process, including the final end points extracted.

**Definition**

- a binary mask $\mathcal{M}$ which defines the region of interest in the image;
- a penalty image $\mathcal{P}$ which will drive the front propagation, usually this penalty map is computed using the centering method described in section 2.3;
- the action map $\mathcal{U}$ computed with this penalty during the initial multiple path extraction;
- the starting point $p_0$, located at the root of the tree hierarchy;
- the set of end points $e_i$ $i \in [1; N_e]$ where $N_e$ is the number of end points extracted.
- a distance step $d$ which defines the minimum distance between two trajectories (this distance step is chosen bigger than the gradient descent step).
- another different label map $\mathcal{L}$ to label the voxels that are neighbors of a path, which means that the distance between this voxel $(i, j, k)$ and a path extracted is less than $d$;
- an array $\mathcal{E}$ to recall the branches detected.

**Initialization**

- $\mathcal{L}(i, j, k) = -1$ for all voxel in the image domain;
- $n_e = N_e$ and $\forall i \in [1; n_e]$, $\mathcal{E}[i] = 0$.

**Loop**: for $i \in [1; N_e]$

- we back-propagate from $e_i$, on the action map $U$ using a simple gradient descent method, as described in equation 2.6;
- at every path step, the position of the new path point is defined by $(x, y, z) \in \mathbb{R}^3$
- we consider the vertices of the Cartesian grid that surround $\overrightarrow{x} = (x, y, z)$, the voxels $\overrightarrow{n} = (i, j, k) \in \mathbb{N}^3$ which verify $D_2(\overrightarrow{x}, \overrightarrow{n}) < d$, where $D$ is the Euclidean distance;
- if, for all those vertices $(i, j, k) \in \mathbb{N}^3$, $\mathcal{L}(i, j, k) = -1$, we set $\mathcal{L}(i, j, k) = i$, and continue back-tracking for $e_i$;
- else, if one of the vertices $(i, j, k)$ verifies $\mathcal{L}(i, j, k) \neq -1$, a branching point is detected, then:
  - recall the label $l = \mathcal{L}(i, j, k)$;
  - $n_e = n_e + 1$, $e_{n_e} = (i, j, k)$;
  - $\mathcal{E}[i] = e_{n_e}$ and $\mathcal{E}[n_e] = 0$;
  - stop back-tracking for $e_i$;

  - continue back-tracking, this time for $e_{n_e}$, substituting all $\mathcal{L}(i,j,k) = l$ by $\mathcal{L}(i,j,k) = n_e$, until another branching point or $p_0$ are found;
- if we reach $p_0$, then stop back-tracking for $e_i$.

**Termination**

- for all end point $e_j$ $j \in [1; n_e]$, we can consider the couples of endpoints $(e_j, e_{\mathcal{E}[j]})$ as extremities of linear parts of the skeleton (with $e_0 = p_0$).
- the multiple paths between couples of points $(e_j, e_{\mathcal{E}[j]})$ $j \in [1; n_e]$ build the skeleton of our object, at scale $c_d$ and distance $d$.

Figure 8.15 displays the result obtained on the dataset shown in figure 8.9. From

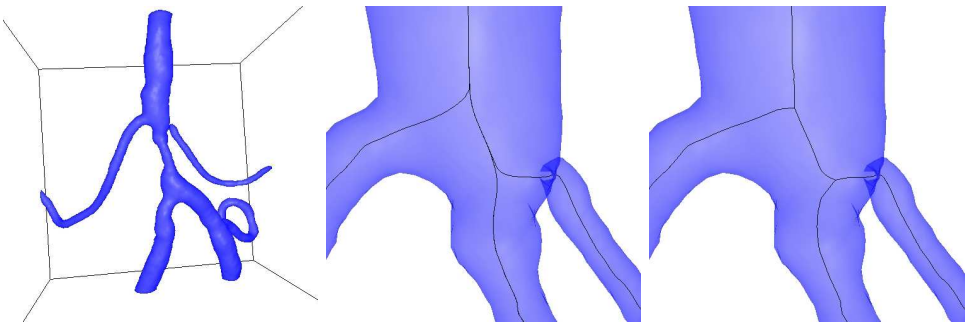


**Figure 8.15. Obtaining a tree hierarchy from a set of trajectory**: Left image is the segmented object extracted from the dataset shown in figure 8.9; middle image is a zoom on two bifurcations of the object, where the trajectories extracted are displayed; right image is the same point of view on the translucent surface extracted with the tree extracted from the set of paths.

the set of multiple trajectories, branching points are extracted, as shown in figure 8.15-right.

### Measurements on the tree

The computational cost of the tree extraction finds its justification in the improvement of the measurements along the new set of trajectories available. Figure 8.16 compares the section measurements with multiple path extraction technique, and tree extraction technique (dataset shown in figure 3.12). The tree extraction, as shown by transparency on figure 8.16-(e) enables to measure the section along the necessary subset of the object, delimited by the the two branching point (this subset has been colored in green on the figure). If this information is plotted across a trajectory in the entire object, it is not useful for two reasons

- section information is not valuable at the branching points;
- the position of the part of interest cannot be obtained straightforwardly.

This problem is illustrated in the last row of figure 8.16. The plot of the object section across the curvilinear abscissae of a trajectory is shown in figure 8.16-(g), versus the same plot across a branch of the tree extracted in figure 8.16-(h).

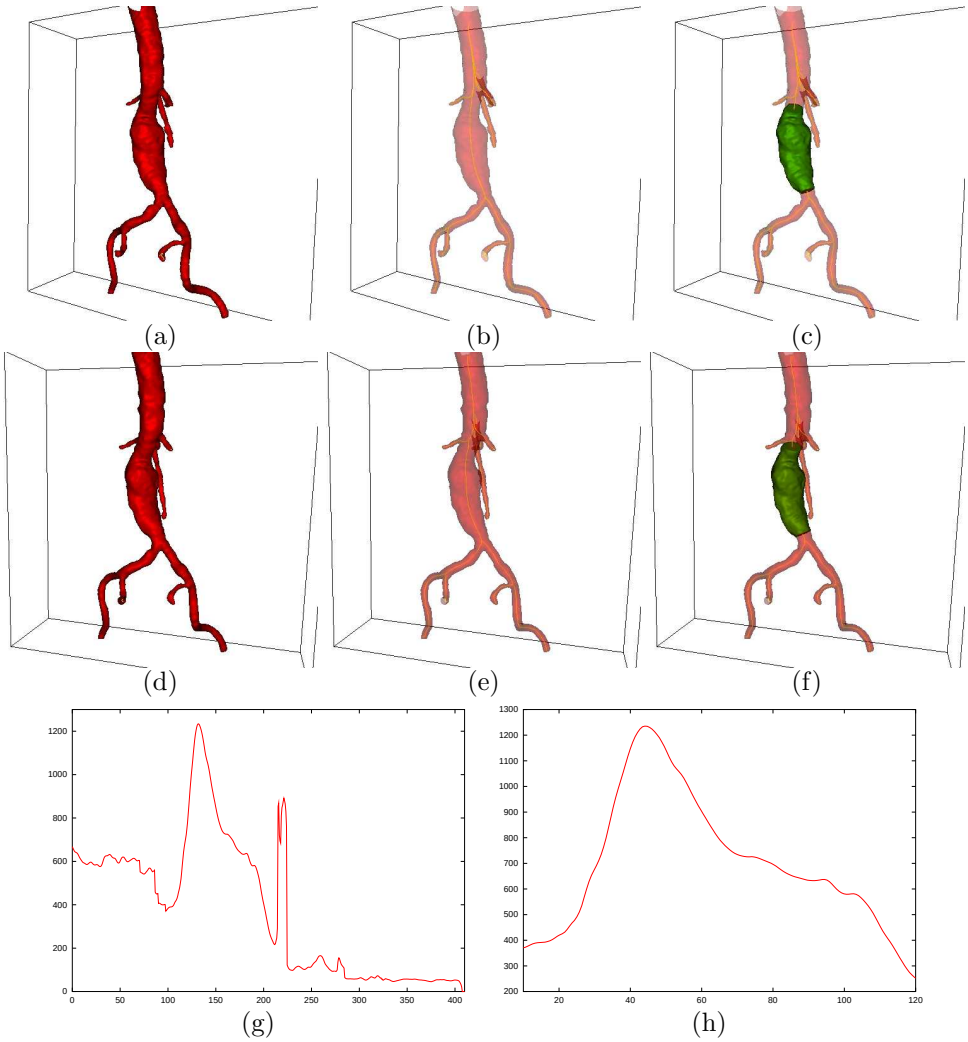



**Figure 8.16. Comparing results with the multiple path extraction, and the tree extraction**: First row shows images of the segmentation process (a) plus the multiple paths extraction, visible by transparency (b) and the region of interest in green that isolate the aneurysm (c) along one of the trajectories; second row shows the same images (d,e,f) using the tree structure extracted from the same seed point; last rows shows the variation of the section along the paths that are inside the aneurysms, for the complete trajectory (g) and for the branch (h).

# Chapter 9

# Application to segmentation and visualization of tree-shaped anatomical objects

**Résumé** — A partir des outils de visualisation et de mesure du chapitre 7, et des outils d'extraction de surfaces et de sequelettes du chapitre 8, nous étudions plusieurs cas pratiques, dédiés à des objets particuliers. Dans la section 9.2, nous appliquons notre méthode d'extraction de trajectoires et d'arbre du chapitre précédent à l'extraction et la quantification des bronches dans les images médicales 3D de scanners CT. Dans la section 9.3, on considère un tout autre problème: celui du groupement perceptuel, ou la donnée est un ensemble non-structuré de régions de l'image. Nous proposons une méthode de reconstruction de structures arborescentes dans des images médicales tridimensionnelles.

**Abstract** — Using the several techniques developed in chapters 7 and 8, we develop applications for medical problems. In section 9.2, we apply the complete multiple paths and shape extraction framework of chapter 8 to the segmentation and quantification of airways in 3D multi-slice CT scanner images. Finally, in section 9.3 we consider the problem of *Perceptual Grouping* and contour completion, where the data is an unstructured set of regions in the image. We propose a new method which is illustrated on reconstruction of tree structures in 3D angiography images.

## 9.1 Application to 3D Vascular Images with Multiscale Vessel Enhancement

In this section, we focus on vascular tree extraction, for accurate determination of vessel width (important in grading vascular pathologies, such as stenosis, or aneurysm). We are particularly interested in using Multiscale Vessel Enhancement techniques of *Frangi et al.* [60].

### 9.1.1 Medical relevance

All methods developed in this chapter are illustrated on the particular problem of vascular tree extraction in 3D contrast enhanced medical images. The medical interest of this extraction is mostly accurate determination of vessel width. It is an important step in grading vascular pathologies, such as stenosis, or aneurysm.

#### Stenosis quantification

In the carotid arteries, this quantification determines the choice of stroke treatment. Studies have revealed that patients with severe symptomatic stenosis in the carotids should undergo surgical treatment, and support the relevance of accurate measurement techniques of vascular segments.

#### Aneurysm quantification

For explanations on this pathology we refer to section 6.1 where they are studied in the case of cerebral vessels. Those pathologies, which are roughly speaking "inflations" of an artery that weak its walls, and can lead to an hemorrhage, occur for example in the brain and, and in the abdominal aorta (see figure 3.12).

#### Potential: Multiscale Vessel enhancement

For the definition of the speed function for the *Fast-Marching* algorithm, we can use the output of a multi-scale vessel filters based on the Hessian matrix [60, 105]. This paragraph will be illustrated by an application on the dataset shown in figure 9.1[1]. We have used the measure defined by *Frangi et al.* [58] in the following. The symmetric Hessian matrix $\mathbf{H}$ describes local second order intensity variations in the image and is given as:

$$H_{ij} = \frac{\partial^2 I}{\partial x_i \partial x_j} \ , \ i, j = 1, \ldots, n \tag{9.1}$$

where $I(\mathbf{x})$ is the n-dimensional image. The Hessian matrix defines an ellipsoid where the direction of its smallest axis is the direction of minimal second derivative, that defines the local direction of a tub-like structure. Having extracted the three eigenvalues of the Hessian matrix computed at scale $\sigma$, ordered $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$, we define

[1] We would like to acknowledge Dr Wiro Niessen, from Image Sciences Institute, University Hospital Utrecht, Netherlands, who provided this image.

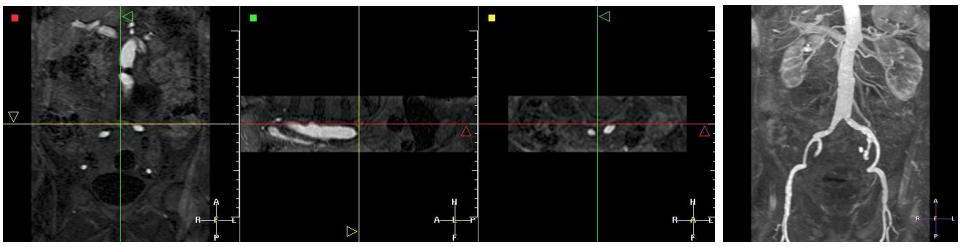


**Figure 9.1. Contrast (Gd-DTPA) MRA image of the aorta**: Left image shows three orthogonal views of the dataset; right image is a **MIP** view.

a vesselness function

$$\nu(s) = \begin{cases} 0, \text{ if } \lambda_2 \geq 0 \text{ or } \lambda_3 \geq 0 \\ (1 - \exp \frac{-{R_A}^2}{2\alpha^2}) \exp \frac{-{R_B}^2}{2\beta^2} (1 - \exp \frac{-S^2}{2c^2}) \text{ else} \end{cases}$$

where the ratios $R_A = \frac{|\lambda_2|}{|\lambda_3|}$ and $R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}}$ are used to distinguish between lines and sheet-like structures and to measure deviation from blob-like structures. These measures arise from geometric interpretation as

$$\begin{aligned} R_A &= \frac{|\lambda_2|}{|\lambda_3|} = \frac{\pi|\lambda_2\lambda_3|}{\pi\lambda_3^2} = \frac{\text{largest cross-sectional area}/\pi}{(\text{largest axis length}/2)^2} \\ R_B &= \frac{|\lambda_1|}{\sqrt{|\lambda_2\lambda_3|}} = \frac{(4\pi/3)|\lambda_1\lambda_2\lambda_3|}{(4\pi/3)((1/\pi).\pi|\lambda_2\lambda_3|)^{3/2}} = \frac{\text{Volume}/(4\pi/3)}{\text{largest cross-sectional area}/\pi^{3/2}} \end{aligned}$$

and $S = \sqrt{{\lambda_1}^2 + {\lambda_2}^2 + {\lambda_3}^2}$ is used to reduce influence of the noise due to intensity variations in the background. See [60] for a detailed explanation of the settings of each parameter in this measure.

In figure 9.2 you can observe the response of the filter, based on the Hessian information, at three different scales: $\sigma = 1, 2, 5$. Using this information computed

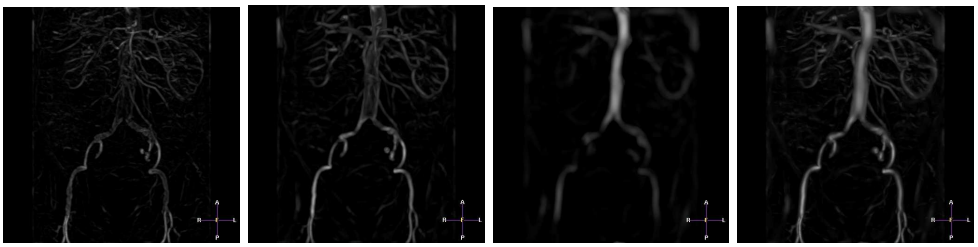


**Figure 9.2. Ridge detection in the aorta image** From left to right, the measure obtained at three different scales ($\sigma = 1, 2, 5$) and the maximum of the filter response across all scales (MIP visualization of the 3D images).

at several scales, the multiscale response of the filter is the maximum of the response of the filter across all scales, which is shown in figure 9.2-right.

Having computed this measure, we simply use it as speed function, thus propagating faster in the areas with higher filter response, taking

$$\|\nabla T(\mathbf{x})\| = \frac{1}{\max_{\sigma_{min}\leq\sigma\leq\sigma_{max}} \nu(\sigma,\mathbf{x}) + \nu_{min}} + w \tag{9.2}$$

where $\sigma_{min}$ and $\sigma_{max}$ are the minimum and maximum scales at which relevant structures are expected to be found, $\nu_{min}$ is just a constant which ensures that speed remains strictly positive, and $w$ is the usual offset term introduced in [34]. This potential will be adapted to bright vascular structures on black background. This potential gives more information than the simple grey level value, since the filter response is higher in the center of the vessel.

### 9.1.2 Proposed solution

**Initialization: *Freezing* method** The use of the *Freezing* improves the resulting segmentation: Figure 9.3 shows the difference of segmentation obtained with (right image) and without pruning (middle image). Figure 9.3 demonstrates that the combination of multi-scale vessel enhancement and freezing enhances the segmentation ability of the *Fast-Marching*.

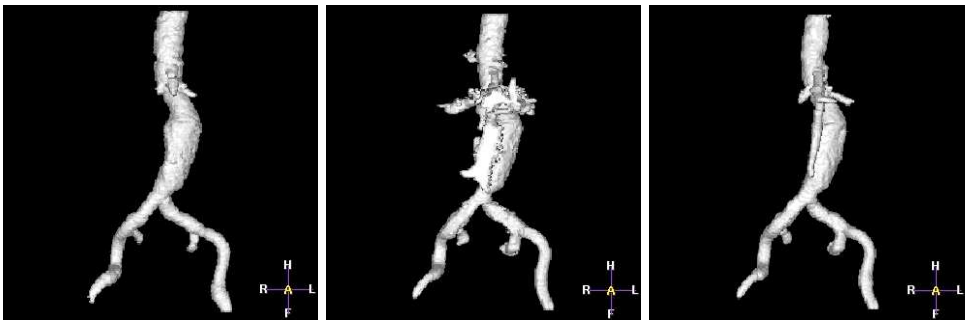



**Figure 9.3. Comparing classical and freezing propagation in the Aorta**: left image shows the resulting volume obtained using the *Fast-Marching* with a penalty model $\mathcal{P}(\mathbf{x}) = \max(I_{mean} - I(\mathbf{x}), 0)$ where $I_{mean}$ is the mean value inside the aorta; middle image shows the result of a wave propagating in the Aorta MR dataset with a speed based on the Hessian eigenvalues; right image shows the same result using the *Freezing* approach of section 8.3.

**Stopping: *Freezing* method** This section is illustrated with figure 9.4[2]. Once the multiscale information is available, we can recall the same data than in the details on the stopping criterion in the previous chapter. If we plot the maximum distance $d_{max}$ of section 8.3.1, across iterations while propagating, we will observe the following profile shown in figure 9.5. A great advantage of this multiscale information is that

[2]We would like to acknowledge Dr Wiro Niessen, from Image Sciences Institute, University Hospital Utrecht, Netherlands, who provided this image.

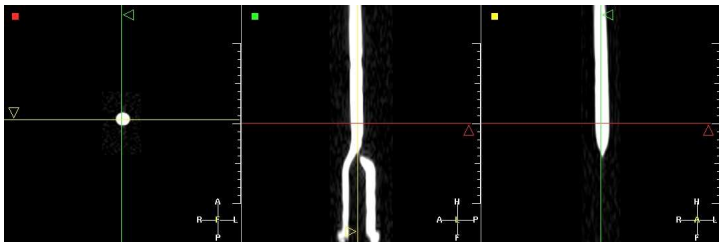


**Figure 9.4. Phantom of a stenosed carotid artery in computed tomography angiography (CTA)**: This example has been chosen for illustration of the stopping criterion, by propagating from the top of the object with a speed $F = 1\ (1 + \|\nabla I\|)$.

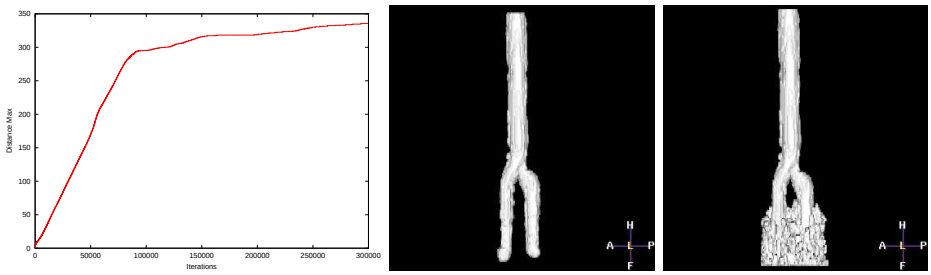



**Figure 9.5. Detecting when the front floods outside the object**: Left image is the plot of the maximum distance $d_{max}$ across iterations; it is clearly visible that there is is an import change in the slope of the function around a distance maximum of 300; middle image represents the domain of voxels visited when $d_{max} = 290$, and right image is the same for $d_{max} = 310$.

it can be used as a potential for obtaining centered trajectories, with no need to compute the distance to the object walls.

### 9.1.3 Comparisons and conclusion of the tree extraction method

#### Other methods for skeletal representation

In [179], authors build a skeletal representation of an unorganized collection of scattered data points lying on a surface. They capture branching shapes, using a distance step similar to ours, by computing the k level-sets from the user-defined root of the tree; and for each of those level-sets, they extract the centroids of connected components. In our case it is not necessary to extract the centroids, because it introduces uncertainty in the location of the branching points. With a centering penalty $\mathcal{P}$, we *aggregate* the paths that are under a user-chosen distance $d$. This method based on the centroid extraction can be compared to the very interesting work of *Angella* found in [5,6], which present a deformable and expansible tree as a skeleton extractor, where each node of the tree is a free particle that propagate into the data, pushed by repulsive forces coming from other particles and contours. The set of free particles

describes the tree hierarchy. In our case, the sub-voxel precision is very important for visualization and measurements (see sections 7.1 and 7.2), and the needed number of particles for achieving this task would lead to huge computing times.

Very similar work can be found in [157], where the author use wavefronts to extract morphological descriptions of binary images, in particular binary tree structures. However, bifurcations are detected on a projected image in 2D, then this information is upgraded to 3D, but still the method is applied iteratively, looking for bifurcations at each iteration. Our method use a scale parameter $c_d$, as a distance step in our wavefront, only looking for bifurcations every time the front has crossed a multiple number of this distance. It reduces greatly computations, and can be parameterized by the user, who can only look for branches lower than a typical value $d_{min}$ which is the upper-bound of our scale parameter.

Morphological techniques, like those in [157], are the main tool used for tree extraction, and lots of techniques, like thinning algorithms are already used in medical imaging. They start from volume images so that the traditional medial axis transform of *Blum* [15] can be applied, as in [130, 143]. However, the purpose of our application is to have a smooth set of multiple trajectories. This smoothness is needed for accurate measurements and visualization along the trajectories. Morphological techniques require post-processing in to remove undesirable small parts of the skeleton. Smoothing and removing undesirable small parts of the skeleton is done using our distance step and is very similar to techniques shown in [173], where the scale is also an input in the algorithm. To conclude with the use of morphological techniques, the skeletal description we are looking for corresponds to the needed of an accurate basis for observation and measurements of pathologies. We thus need a smooth and accurate information: a tree which describes the cylindrical topology of the object observed. The variation of the section of a tubular shapes leads to error in medial axis transforms, and to the need of post-processing techniques, to *clean* the skeleton obtained, that our method does not need.

Most impressive work on vascular quantification among others can be found in the PhD thesis of *Frangi*. He develop a very interesting method based on path and shape extraction in [58]:

- The author first set the two extremities of a path on the surface obtained through a iso-surface extraction process;
- the minimal path is extracted on the representation of the surface, using a technique similar to [90];
- a centering force, based on multi-scale enhancement filtering (see [60]) drives the minimal path in the center of the tube-shaped object;
- a circular cross section approximating the vessel is swept along the central vessel axis extracted previously (*swept surface*), and creates a deformable cylinder;
- this cylinder initiates a tensor product B-spline surface [142], that fits the boundaries of the vessel.

Using both path and shapes representation in the same framework, Frangi proposes an elegant method for quantification of vessel morphology [59].

In order to highlight important benefits of our method, we are going to compare our results, that are not completely dedicated to quantification of vascular diseases in MRA images. However, our method proposes an alternative that may overcome several drawbacks of his method.

**Topology of the objects:** In [57] the bifurcations in carotid arteries introduce errors in the measurements of the stenoses; with our method, bifurcations are localized and wrong measures near branching points can be omitted.

**Branching points:** We provide the measures in the whole set of branches of our objects, setting a unique tree root seed for segmentation and path extraction. In [57] *Frangi* gives the measure between the defined user end points (he gives also an interesting study of the variability of the results across the user initialization in [57]). In our case, only one point is needed. It enables to reconstruct the whole set of trajectories inside the object, but it converts this information into a tree hierarchy, where important information can be separated from the whole.

A result of this property of our method is shown in figure 9.6. In particular, figure 9.6-(g) is the information contained in the interval $[40; 60]$ in figure 9.6-(e). It is the same process for figure 9.6-(h) which corresponds to the sub-plot contained in the interval $[190; 250]$ in figure 9.6-(e). Therefore the tree extraction enables to localize accurately the information needed, as the stenosis extent for the case presented in the left column of figure 9.6.

**Accuracy of the model** the B-spline that extracts the vessel boundaries in [58] is an approximation of the surface, whereas the zero-level set embedded in $\tilde{\phi}$ has sub-pixel precision.

**Conclusion on the vascular extraction method**

We have finally a method which provides a sub-pixel information of the position of the shape. Based on the paths extracted with our fast and robust algorithm, the quantification rely on an accurate centered position of the path points. Thus measures and visualization are enhanced (see figure 9.7).

At a matter a fact, this visualization, once paths and shapes are extracted, is real-time, due to the fast rendering of the triangulation of our implicit model. Thus, camera trajectory is managed via the paths extracted. A further extension of this work could be to derive an interface to choose between each branch where to go inside the model.

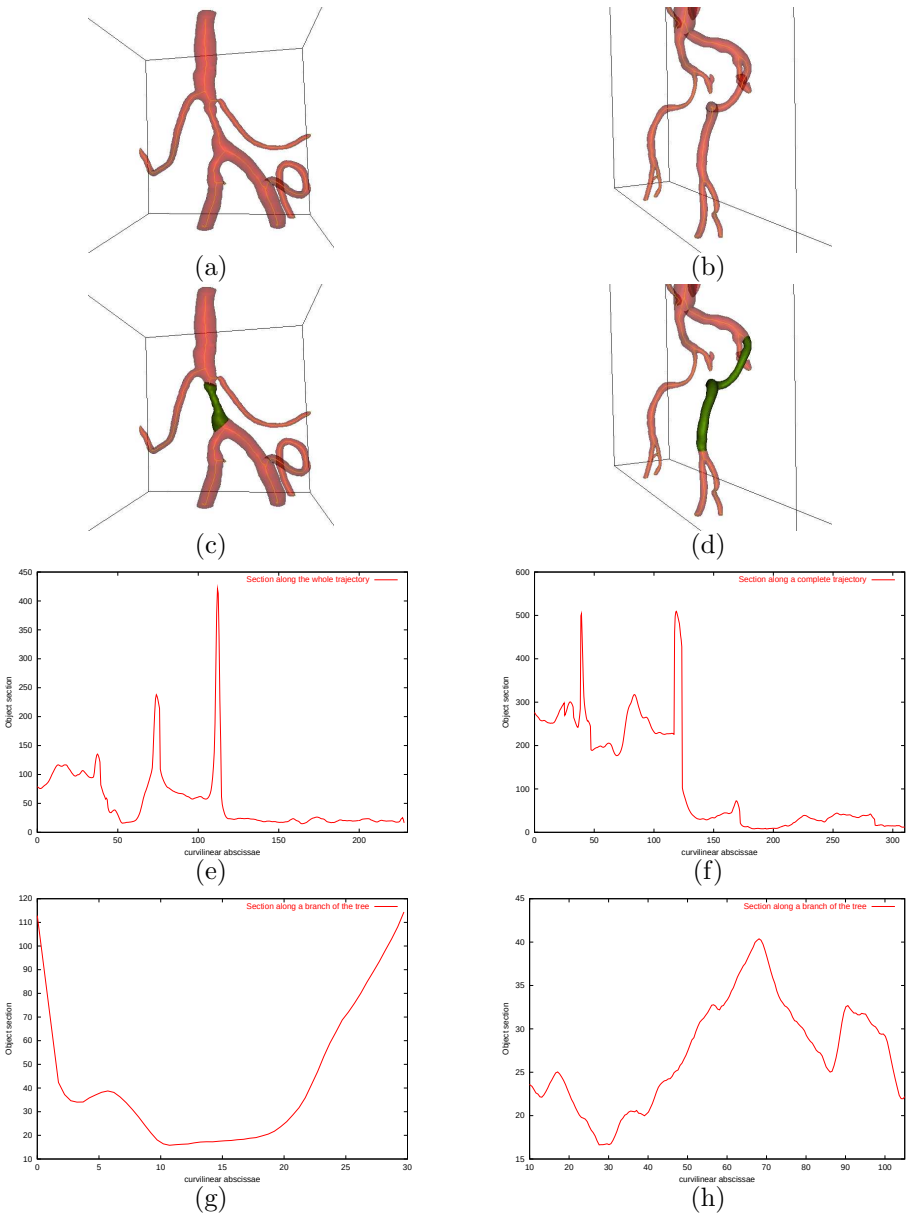



**Figure 9.6. Comparison of measurements on the tree and on a trajectory**: Two different datasets are presented, each one in a column (left column dataset present a stenosed vessel). First row (a,b) displays segmented surfaces and extracted trees. Second row (c,d) displays the sub-volume of interest in both cases where sections are performed. Third row (e,f) shows plots of the section measured across the curvilinear abscissae of a trajectory. Fourth row (g,h) displays the same result using branches extracted between two bifurcations.

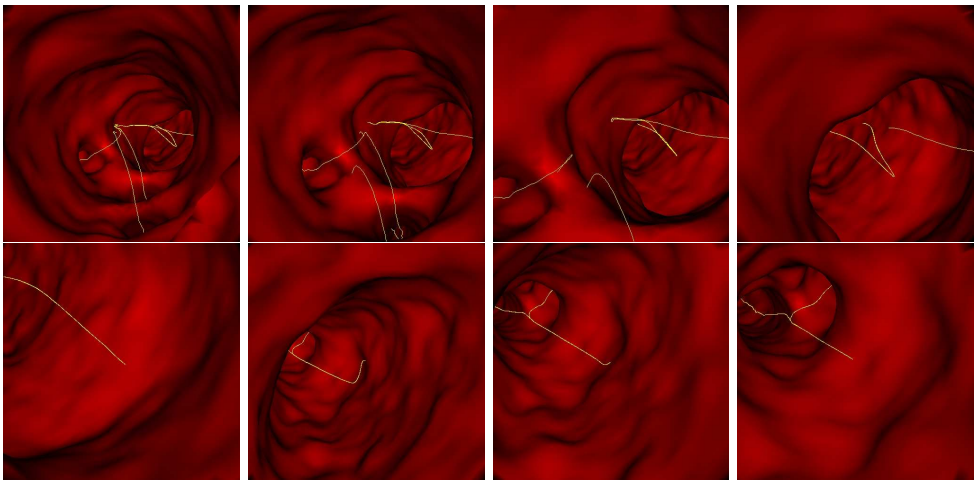


**Figure 9.7. Endoscopic view along one trajectory**: The whole set of trajectories is displayed (in yellow) simultaneously with the surface rendering: the user do not miss any bifurcation since he can see any branch.

## 9.2 Application to the Bronchial Tree

### 9.2.1 Medical interest

#### Extraction of the bronchi and the bronchial tree

First interest of this segmentation is to provide trajectories for a virtual bronchoscopy system, on the same basis than the virtual colonoscopy tool detailed in section 3.1.

A second important implementation possible is airway tree measurements that can be used to detect lesions or stenoses, structural abnormalities, and to evaluate airway reactivity to external stimuli (for example, evaluating asthma impact on the airways diameters). One must take into account the perspective distortion inherited from the volume rendering, for optimizing stent fitting for example, and the segmentation of the complex bronchi can provide an accurate information.

The framework we developed for the vascular tree extraction can be transposed to this different problem, in order to obtain both surfaces and centerlines of the airways, in typical datasets of the lungs, like multi-slice CT scanner, where voxels are nearly isotropic.

#### Role of the Virtual Bronchoscopy

The bronchoscopy technique has existed since 1897 and represents probably one of the most frequently used invasive procedures. Even in the hands of a clinically experienced pulmonologist, there is a risk for the patient. However, the goal of virtual bronchoscopy (see an example in [75]) is not to replace real bronchoscopy, which has high advantage of providing a direct inspection of the natural pigmentation which

can clearly indicate a pathology. And inability to perform biopsies gives virtual bronchoscopy the role of a detection and much less a characterization technique.

Possible pathologies visible on the multi-slice CT scanner are for example tumors. Malignant tumors of the lung represent the most frequent cause of cancer death in males (35%) and female (18%). Since those tumors (benign or malignant) are only visible on the renderings system when they imply a morphological alteration of the bronchial wall, virtual bronchoscopy cannot contribute substantially to characterization of tumors. Thus, evaluation of virtual bronchoscopy should be restricted to the morphology of the bronchi and direct visualization of intraluminal masses.

However, one handicap of real endoscopy is its inability to see through the bronchial wall, whereas all information surrounding the object is available in virtual bronchoscopy. This handicap is very important since a clinician would like to plan a biopsy in a location that can be accessed only from the bronchial tree. In this case, the virtual bronchoscopy enables to determine in advance the optimal access point for the biopsy procedure.

A further indication for this process is the rare case when the real inspection is contraindicated, as in the presence of a strong stenosis of a branch, or as in the presence of an infiltration due to an extensive tumor manifestation, or in the case of an application in pediatrics, where the necessary sedation can be contraindicated.

Last important improvement brought by the virtual procedure is its clinician teaching device aspect [18, 125]: it can contribute th education and qualification of operating personnel (which benefits the patient by the way).

### 9.2.2 State of the art in Bronchoscopy imaging

#### Acquisition techniques

Computed Tomography represents the standard examination technique of the thoracic area, because a natural contrast exists between air and soft tissues, explaining why the trachea and the bronchial tree are perfectly suited for the generation of a virtual bronchoscopy. Three different types of CT data can be used:

1. Incremental CT: a slice is imaged in the axial orientation, after which the patient is shifted to the next position in order to image the next slice. 3D reconstruction (and hereafter renderings) can be calculated from incremental data only when the patient lies so still than no motion occurs during the whole examination. These data are of little use for virtual bronchoscopy;

2. Spiral CT: superior to incremental CT, the patient is shifted during the rotation of the tube-detector system. It enables to acquire large anatomical regions, like Thorax, in a single breath-hold. But still, the z-axis resolution is considerably worse than the in-plane resolution, and is a limiting factor for small bronchi.

3. Multi-slice CT: common systems image four slices at a time. This results in very low total acquisition time, but can also result in isotropic volume elements (voxels) in the final 3D dataset. Figure 9.8 displays a volume of interest of a classical multi-slice CT scanner of the lungs.

**Figure 9.8. 3D Multi-Slice CT scanner of the Lungs**: this is an isotropic volume of interest about $287 \times 150 \times 249$ voxels.

Multi-slice CT is the ideal imaging modality for obtaining virtual bronchoscopic renderings of the smaller bronchi, since structures with diameter of 2mm should become clearly visible. However, the huge number of voxels can create a problem, as it overload the storage capacity of common workstations. Processing time in algorithms can also become so long than clinical application may not be realistic.

### Segmentation techniques

They can be divided in three categories:

1. 2D methods: they can be completely manual, where the contours of the airway lumen is delineated manually in each axial images, but they strongly depend on the interobserver variability, and the loss of precision of the 2D segmentation towards the 3D information. They can also be semi-automatic, starting from an initial set of contours manually drawn, which is corrected and smoothed with detection algorithms, or by flooding gradient maps with region-growing in 2D [144]. However, those 2D methods work on the information contained in the cross-sections of the bronchi contains on not on the whole 3D data.

2. 3D methods: there is of course the usual set of 3D rendering techniques (**MPR** , **MIP** , volume or surface renderings). Other more elaborated methods are based on 3D region-growing algorithms. *Mori et al.* [124] detect an optimal threshold value in order to extract the airways, and once this segmentation is done, thin the airways to obtain the tree which is input in a recognition process for automatic labeling [123]. Those methods suffer from limitations due to the use of a threshold, and problems of accuracy for the small bronchi, since they provide a binary image as segmentation.

3. 2D to 3D methods: the principle is to segment the airways in each axial slice, and to reconstruct the 3D segmentation by combining the 2D segmentations, doing 3D/3D post-processing. Several techniques are based on the detection of the airways location in 2D [169]. But how important is the enhancement of the post-processing, the result depends on the 2D initialization, there is no pure 3D information involved in the detection. However, important improvements have been done in the field by *Fetita* [52], as well in 2D/3D as in pure 3D.

We have worked upon the use of our method involving both *Fast-Marching* and *Level-Sets* methods to extract the airways, and the airway tree. We encountered several problems due to the specificity of the bronchi in CT, during the initialization step. All tests are detailed in the following.

### 9.2.3 Applying our framework

#### Initialization: Region growing

Several problems exist with the use of the *Fast-Marching* algorithm as a region growing method for airway segmentation. Since it relies on the edge strength of the airway walls that weak at several places, the propagating front *leaks* out into the surrounding parenchyma. We have already seen that the *Fast-Marching* can flood into the surrounding pixels of the tubular structures, and we have built a method based on *Freezing* pixels (see section 8.3.1) in order to avoid *leakage*. This method was successfully applied for vascular tree extraction in chapter 8, using as speed function a multiscale filtering technique derived from work in [60]. But in the case of the lungs airway, the grey level information is very different, as shown in figure 9.9. In fig-

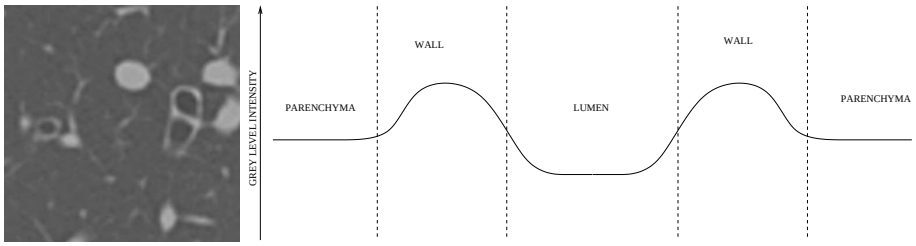



**Figure 9.9. Profile of the airway grey level**: Left image is a zoom on a 2D axial slice of the multi-slice reformat image of the lungs, we can distinguish the lumen in the black circle areas surrounded by the airway walls in bright intensities; right image is an illustration of the profile of the grey level intensity, along a line crossing through an airway center.

ure 9.9-right, we observe that the minima occur near the center of the airways and the maxima occur near the middle of the walls. But this information is relatively poor, and partial volume effects can occur: as the diameter of the airways decreases, partial volume averaging begins to increase the value within the lumen, and the *Fast-Marching* will flood the parenchyma at a weak wall, as shown in figure 9.10. The *Fast-Marching* algorithm is applied in this case with a potential based on the grey level information. Using $\mathcal{P}(\mathbf{x}) = \max\left(I(\mathbf{x}) - I_{airways}, 0\right) + w$, with $I_{airways}$ being an approximate value corresponding to air, cannot provide a result where the propagation critically depends on the weakness of the edges. Using the grey level information for the lungs is similar to using the gradient in the vascular contrast enhanced medical images: it is not valuable. The profile of the airway in figure 9.9-right is somehow similar to a "Mexican hat", but the Hessian information given by a measure based on its eigenvalues will detect the inner and outer walls of the airways, and will not give a high response in the lumen. Since then, this problem defines the limits of the use of

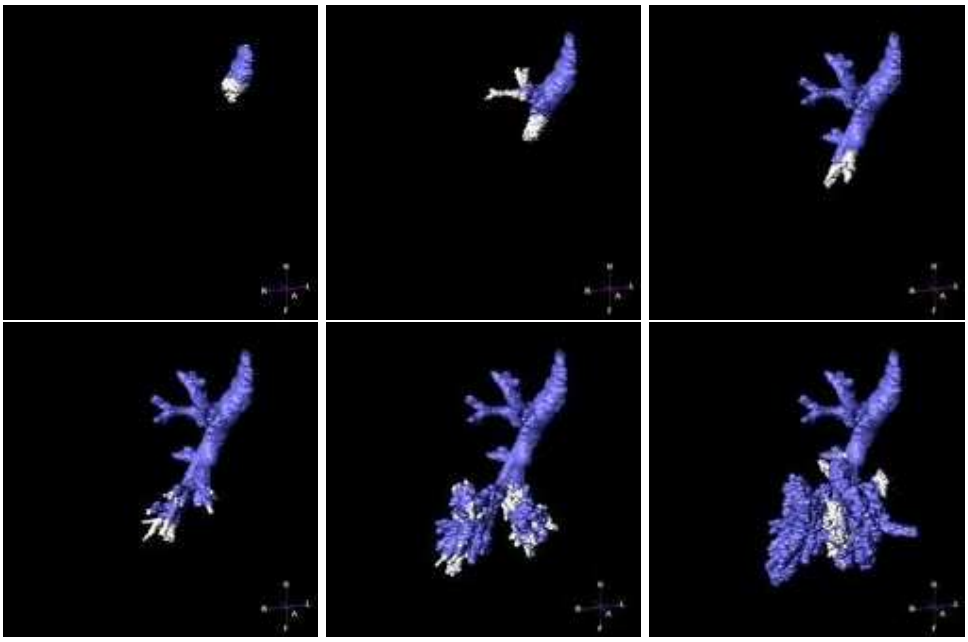


**Figure 9.10. Flooding of the *Fast-Marching* in the parenchyma**: the images are samples of the front propagation process in the lungs airways, with the *Freezing* methodology; the frozen pixels are represented in blue whereas the propagating parts are in white.

the *Fast-Marching* in pre-segmentation, as well as with the use of a more complicated heuristic such as the *Freezing* algorithm.

However, there are other ways to create a pre-segmentation which can be input in a *Level-Sets* model. Among others, we decided to use methods developed by *Mori et al.* in [124]. The principle of this initialization is based on the reason of the failure of the *Fast-Marching* method: it focuses on the detection of the *flooding* inside the parenchyma. It extracts the inside of the area of the bronchus by tracing voxels with relatively small CT values corresponding to air without processing across voxels with relatively large CT values, assuming that the airway area is simply connected.

The method is a simple region-growing in the 3D image, starting from a point inside the trachea. This point, which will be the root of the final tree hierarchy can be easily detected in the 3D dataset, as shown in figure 9.8

The algorithm is the following:

**Definition**

- a start point $\mathbf{x}_0$: the region growing needs a seed point for starting. In order to set a protocol of segmentation, the start point needs to be always initialized in the same region, inside the trachea, before the first bifurcation in the tracheobronchial tree. Luckily enough, the trachea can be recognized stably in the multi-slice CT dataset. Moreover this seed point will be the root of the tree hierarchy extracted at the end of the whole process, which enables to reduce user interaction to the setting of the seed

point only.

- an initial threshold value $I_0$, a threshold value $I_{threshold}$ with a threshold step $dI$, for the original volume image $I$
- a segmentation defined by a binary mask $\mathcal{M}$ where $\mathcal{M}(\mathbf{x}) = 0$ if $\mathbf{x}$ is inside the object, and $\mathcal{M}(\mathbf{x}) = -1$ elsewhere.

**Initialization**

- $I_{threshold} = I_0$;
- $\mathcal{M}(\mathbf{x}_0) = 0$ and $\mathcal{M}(\mathbf{x}) = -1$ elsewhere;

**Loop:**

- at each iteration $i$, we binarize image $I$, defining the mask $I_B$ where $I_B(\mathbf{x}) = 0$ if $I(\mathbf{x}) < I_{threshold}$, $I_B(\mathbf{x}) = -1$ elsewhere;
- we apply a connectivity algorithm, to connect to the pixels $\mathbf{x}$ that verify $\mathcal{M}(\mathbf{x}) = 0$ all voxels $\mathbf{y}$ with $I_B(\mathbf{y}) = 0$.
- For all voxels $\mathbf{y}$ connected, $\mathcal{M}(\mathbf{y}) = 0$;
- We count $N_i$ the total number of voxels $\mathbf{x}$ with $\mathcal{M}(\mathbf{x}) = 0$ at iteration $i$ (see figure 9.11);
- If $N_i > N_{max}$, the optimal threshold is $I_{threshold}$ and we stop;
- $I_{threshold} = I_{threshold} + dI$.

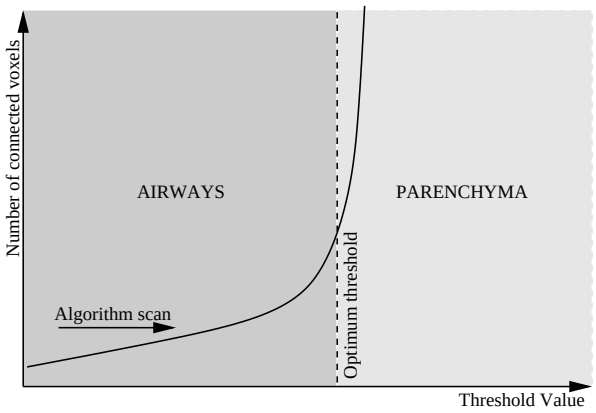



**Figure 9.11. Detection of the optimal threshold value**: When the threshold value is increased, the number of connected voxels below this threshold increases, until it reaches the optimal value; a superior threshold will lead to the flooding in the parenchyma.

When the number of segmented voxels exceeds the area threshold value $N_{max}$, the algorithm is stopped, just before explosion in the number of voxels visited occurs in figure 9.11.

**Final Segmentation and Automatic Branch extraction**

Since the initialization is obtained, the inside and outside regions are used in order to initialize the region-based forces of the *Level-Sets* model. In the case of the lungs, we used the sigmoidal region-based forces formulation for several reasons:

- the computing time is very important, considering the size of the dataset, and the use of constant forces across time reduces this cost;
- the parenchyma has a distribution very similar to that of the airways. The use of region-based forces should then induce the use of three different regions: one for the lumen, one for the parenchyma, and one for the soft tissues (and others). Managing three regions will greatly increase the computing time.
- if we use two regions instead of three, the parenchyma is contained in the outside region and reduces greatly the mean and increases the variance of the model. There is a huge risk that the outside has a variance too important, and shrinks the lumen segmented. This problem has been already presented in the application concerning the visualization of the colon polyps.

Several iterations are necessary in order to extract the lower bronchi, as shown in figure 9.12. Once this segmentation step is achieved, we extract the trajectories from

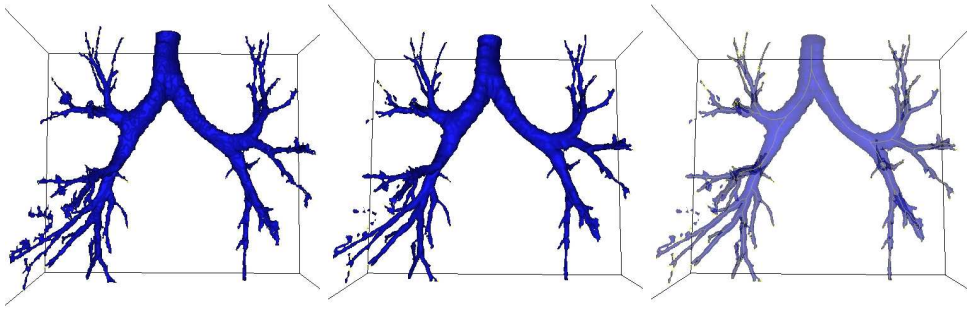


**Figure 9.12. Three steps of the airways tree segmentation**: Left image is the initialization given by the method [124] described in section 9.2.3; middle image is the surface of the airways after 40 iterations of the *Level-Sets* model; right image shows the whole tree extracted in the airways with the labeling algorithm illustrated in figure 9.13.

the starting point $\mathbf{x}_0$, and we convert it into a tree, using the labeling methods of section 8.4. We optimize this extraction, in function of the length of the minimal branch to be extracted, as shown in figure 9.13, where we display the label map for several minimal length. Since this step represents a computing cost, this minimal length must be accurately set according to the needs of the clinician (since this length shrinks with the depth in the bronchial tree).

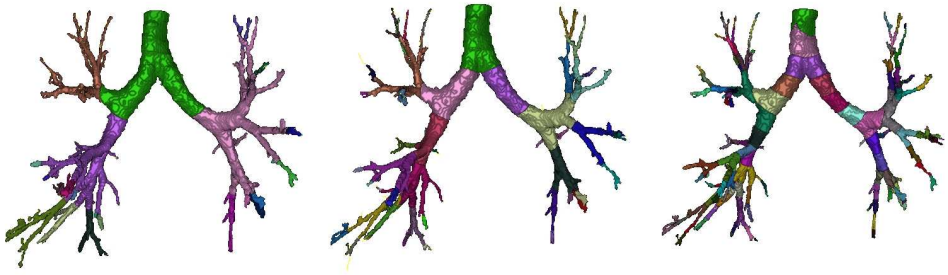


**Figure 9.13. Labeling of the airways**: Using the method described in section 8.4, we label the airways segmented, according to a chosen distance step; from left to right are shown the label maps for respective steps 100, 50 and 20.

### 9.2.4 Conclusion

#### Artifacts

Even with multi-slice technology, one problem remains unsolved: cardiac motion cannot be suppressed, despite fast acquisition. Only a trigger, as employed for cardiac imaging could provide assistance. This motion extends to the neighboring pulmonary parenchyma, and results in irregularities in the bronchial wall (see figure 9.14-left). The impact of this motion onto the segmentation is not clear, but it seems that the

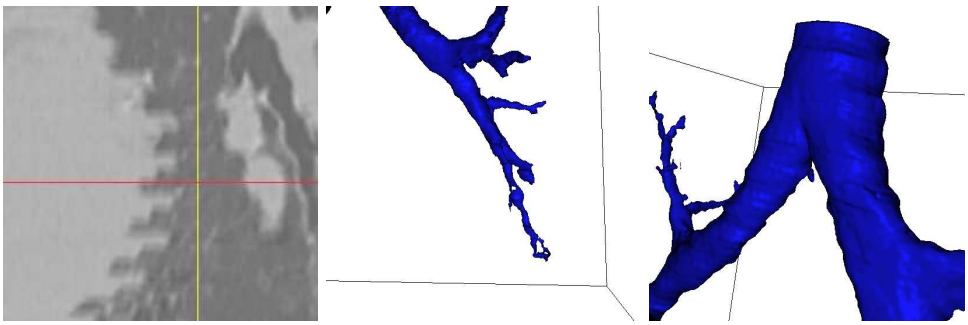


**Figure 9.14. Two kind of artifacts in the multi-slice datasets**: Left image shows the motion of the heart in a sagittal slice of the 3D dataset of figure 9.8 during acquisition; middle image shows the resulting poor segmentation of the lower-left part of the airways; right image displays the pulsation artifacts carried over from the aorta onto the trachea: these rings should not be confused with the natural tracheal cartilage.

surface extracted near the hear in the lower left part of the bronchi is not correctly segmented (see figure 9.14-middle). These artifacts can be reduced by doing multi-slice CT acquisition with a trigger, as done for cardiac imaging, taking into account cardiac pulsation for the time of acquisition.

Much more striking are artifacts produced by the aortic pulsations, transmitted to the trachea and the main bronchi: they appear as ring-like structures on the 3D surface, similarly to cartilage rings. However, they can easily be distinguished from the tracheal cartilages, since they appear horizontal in the lice direction (see figure 9.14-right). Those rings imply errors in the measures of the airways diameters, but there is no particular possibility to avoid them, since a correlation with the heart pulsations exists but is difficult to model.

**Perspectives**

Results are impressive, and we can easily obtain a virtual bronchoscopic view, using the tree structure to guide the virtual endoscope and the triangulated surface, obtained through th *Marching-Cubes* algorithm with the zero-level set of our level-set function (see figure 9.15)

However, there are several technical improvements that are currently missing

- Improving fast-marching for the airway initialization: the method developed has failed in giving an accurate initialization. The complex structure of the object, and the thin-walled bronchi plus the partial volume effect lead to wrong results. One possible extension could be to modify the speed function, adapting it to the depth of the current voxels involved in the computation, since the partial volume effect increases with the depth in the airway tree.

- Reducing the number of iterations of the *Level-Sets* model: 40 iterations is still a huge number if each iterations is computed on the whole volume.

- Improving the *Level-Sets* model: the smaller parts of the bronchi are not recovered, the region-based formulation is not dedicated to the extraction of the thin curves. Using co-dimension 2 geodesic contours, as done by *Lorigo et al.* [108] for vessels in MRA images, and using other expression of the flows like *Vasilevskiy and Siddiqi* [177].

- Developments: *Mori* was using its tree extraction method for automatic labeling of the bronchial tree [123]. If it is possible to assign the anatomical names to the bronchial branches extracted from CT images and to display the name of the currently observed branch on a virtual bronchoscopy image, it will help clinicians to understand the current observing position. This tree structure can also be input in a system to assist biopsies in the tracheobronchial tree, as done in [17].

- Validation: still, clinical evaluation of the method is not done, and in particular the evaluation of the tree structure extracted should be evaluated with the choice of the minimal length of the branches extracted.

- Benchmark: a possibility of a benchmark with the tremendous work of *Fetita* [52, 144] has been scheduled but is not achieved yet.

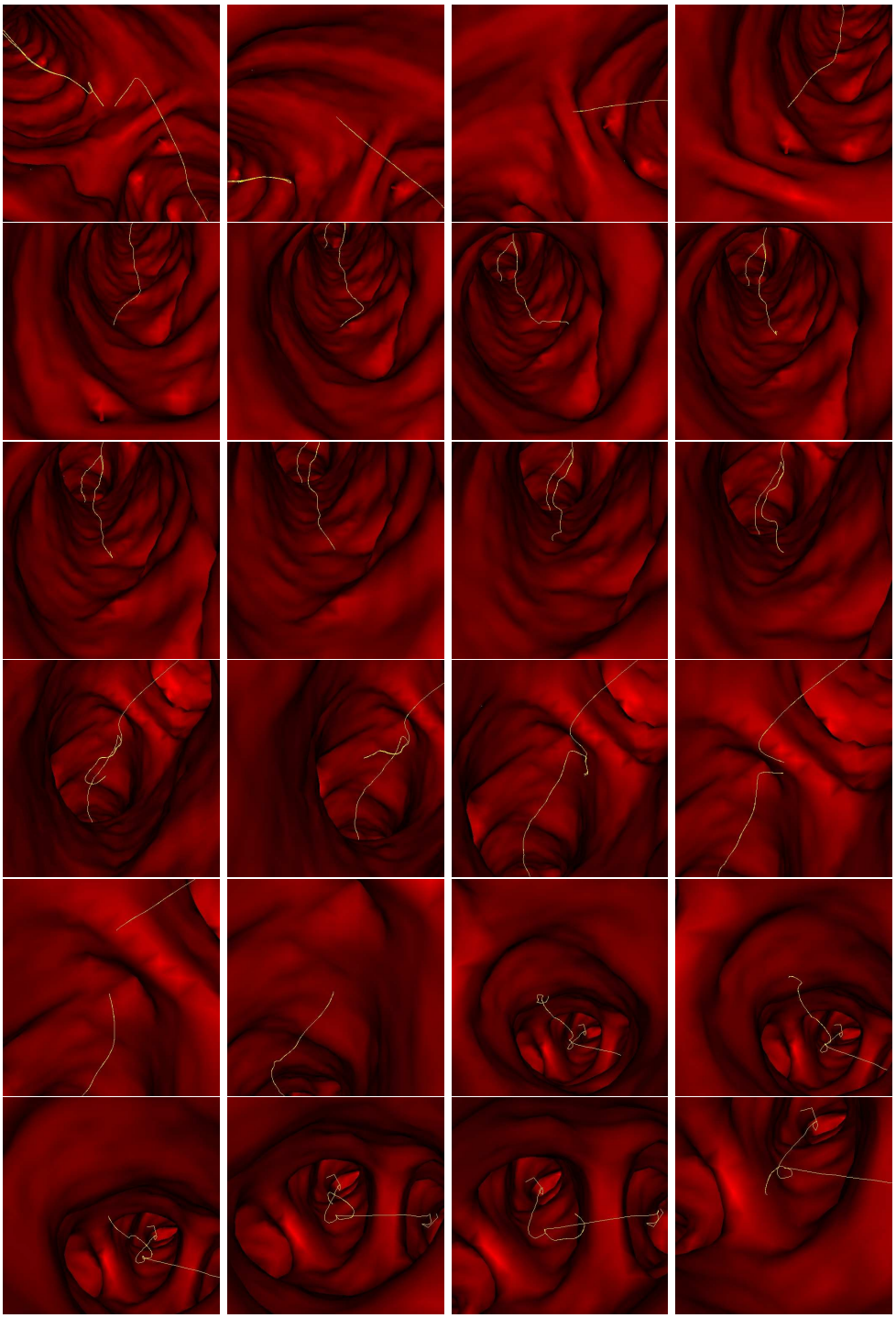


**Figure 9.15. Virtual Bronchoscopy**: the images are samples of a movie automatically generated with the surface and the tree extracted in the 3D dataset with our Path and Shape extraction framework.

## 9.3 Reconstruction of vessels in 2D and 3D images using *Perceptual Grouping*

Since their introduction, active contours [82] have been extensively used to find the contour of an object in an image through the minimization of an energy. In order to get a set of contours with T-junctions, we need many active contours to be initialized on the image. The level sets paradigm [23,113] allows changes in topology. It enables to get multiple contours by starting with a single one. However, it does not give satisfying results when there are gaps in the data since the contour may propagate into a hole and then split into several curves where only one contour is desired. This is the problem encountered with *Perceptual Grouping* when we try to group a set of incomplete contours. For example, in a binary image like in figure 9.16 with a drawing of a shape with holes, human vision can easily fill in the missing boundaries and form complete curves. *Perceptual Grouping* is an old problem in computer vision. It has been approached more recently with energy methods [72, 164, 187]. These methods find a criteria for saliency of a curve component or for each point of the image. This saliency measure is based indirectly on a second order regularization snake-like energy ( [82]) of a path containing the point. However, the final curves are generally obtained in a second step as ridge lines of the saliency criteria after thresholding. Motivated by this relationship between energy minimizing curves like snakes and completion contours, we worked upon finding a set of completion contours on an image as a set of energy minimizing curves.

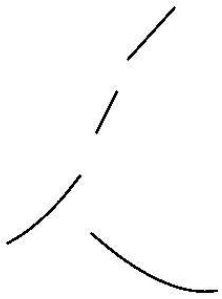


**Figure 9.16. Examples of connected regions to be completed:** The four regions are the four black components, on a bright background.

In order to solve global minimization for snakes, *Cohen and Kimmel* [34] used the minimal paths, as introduced in [86, 87]. The goal was to avoid local minima without demanding too much on user initialization, which is a main drawback of classic snakes [29]. Only two end points were needed. The numerical method has the advantage of being consistent (see [34]) and efficient using the Fast Marching algorithm introduced in [161]. In [31], the author proposed a way to use this minimal path approach to find a set of curves drawn from a set of points in the image. We also introduced a technique that automatically finds a set of *key* end points. In this chapter, we extend the previous approach to connected components instead of end

points. In order to obtain a set of most salient contour curves, we find a set of minimal paths between pairs of connected components.

This approach is then extended for application in the completion of tube-like structures in 2D and 3D images. The problem is here to complete a partially detected object, based on some detected connected components that belong to this object.

For *Perceptual Grouping* , the potential $P$ to be minimized along the curves is usually an image of edge points that represent simple incomplete shapes, as in figure 9.16. These edge points are represented as a binary image with small potential values along the edges and high values at the background. The potential could also be defined as edges weighted by the value of the gradient or as a function of an estimate of the gradient of the image itself, $P = g(\|\nabla I\|)$, like in classic snakes. The potential could also be a grey level image as in [34]. It could also be a more complicated function of the grey level. In our real examples of vascular structures in 2D and 3D, we use a potential based on a vesselness filter [60].

We present in Section 9.3.1 how to find a set of curves from a given set of unstructured points. Grouping the points in connected components, we propose a way to find the pairs of linked connected components and the paths between them. We then extend this approach to 3D and show an application in 3D medical images.

### 9.3.1 Finding Contours from a Set of Connected Components

#### Minimal Path between two Regions

The method of [34], detailed in the previous section allows to find a minimal path between two endpoints. This is a straightforward extension to define a minimal path between two regions of the image. Given two connected regions of the image $\mathbf{R}_0$ and $\mathbf{R}_1$, we consider $\mathbf{R}_0$ as the starting region and $\mathbf{R}_1$ as a set of end points. The problem is then finding a path minimizing energy among all paths with start point in $\mathbf{R}_0$ and end point in $\mathbf{R}_1$. The minimal action is now defined by

$$U(p) = \inf_{\mathcal{A}_{\mathbf{R}_0,p}} E(C) = \inf_{p_0 \in \mathbf{R}_0} \inf_{\mathcal{A}_{p_0,p}} E(C) \tag{9.3}$$

where $\mathcal{A}_{\mathbf{R}_0,p}$ is the set of all paths starting at a point of $\mathbf{R}_0$ and ending at $p$. This minimal action can be computed the same way as before in table 2.1, with the alive set initialized as the whole set of points of $\mathbf{R}_0$, with $U = 0$ and trial points being the set of 4-connexity neighbors of points of $\mathbf{R}_0$ that are not in $\mathbf{R}_0$. Back-propagation by gradient descent on $U$ from any point $p$ in the image will give the minimal path that join this point with region $\mathbf{R}_0$.

In order to find a minimal path between region $\mathbf{R}_1$ and region $\mathbf{R}_0$, we determine a point $p_1 \in \mathbf{R}_1$ such that $U(p_1) = \min_{p \in \mathbf{R}_1} U(p)$. We then back-propagate from $p_1$ to $\mathbf{R}_0$ to find the minimal path between $p_1$ and $\mathbf{R}_0$, which is also a minimal path between $\mathbf{R}_1$ and $\mathbf{R}_0$.

#### Minimal Paths from a Set of Connected Components

We are now interested in finding many or all contours in an image. We assume that from some preprocessing, or as data, we have an initial set of contours. We denote

$\mathbf{R}_k$ the connected components of these contours. We propose to find the contours as a set of minimal paths that link pairs of regions among the $\mathbf{R}_k$'s. If we also know which pairs of regions have to be linked together, finding the whole set of contours is a trivial application of the previous section. The problem we are interested in here is also to find out which pairs of regions have to be connected by a contour. Since the set of contours $\mathbf{R}_k$'s is assumed to be given unstructured, we do not know in advance how the regions connect. This is the key problem that is solved here using a minimal action map.

### Method

Our approach is similar to computing the distance map to a set of regions and their Voronoi diagram. However, we use here a weighted distance defined through the potential $P$. This distance is obtained as the minimal action with respect to $P$ with zero value at all points of regions $\mathbf{R}_k$. Instead of computing a minimal action map for each pair of regions, as in Section 9.3.1, we only need to compute one minimal action map in order to find all paths. At the same time the action map is computed we determine the pairs of regions that have to be linked together. This is based on finding meeting points of the propagation fronts. These are *saddle points* of the minimal action $U$. These saddle points were already used for closed boundary extraction in [34] In Section 1.1.2, we said that calculation of the minimal action can be seen as the propagation of a front through equation (1.5). Although the minimal action is computed using fast marching, the level sets of $U$ give the evolution of the front. During the fast marching algorithm, the boundary of the set of alive points also gives the position of the front. In the previous section, we had only one front evolving from the starting region $\mathbf{R}_0$. Since all points $p$ of regions $\mathbf{R}_k$ are set with $\mathcal{U}(p) = 0$, we now have one front evolving from each of the starting regions $\mathbf{R}_k$. In what follows when we talk about front meeting, we mean either the geometric point where the two fronts coming from different $\mathbf{R}_k$'s meet, or in the discrete algorithm the first alive point which connects two components from different $\mathbf{R}_k$'s (see Figures 9.17 and 9.18).

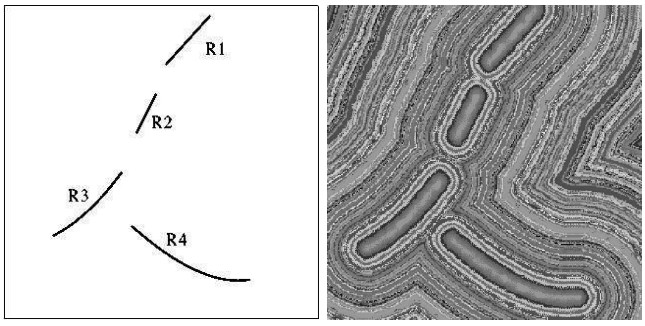



**Figure 9.17. Minimal Action map from the four regions of the example of figure 9.16:** On the right with a random LUT to show the level sets.

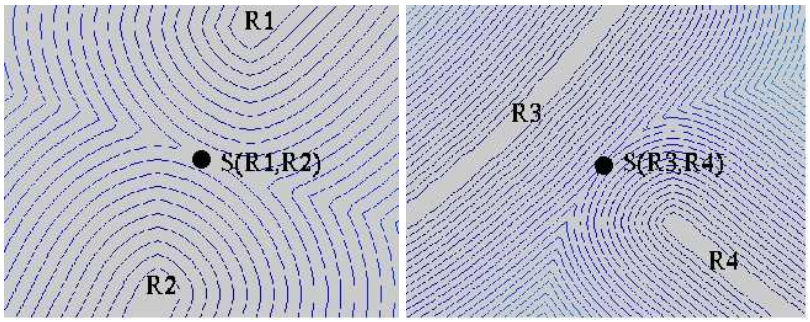



**Figure 9.18. Zoom on** *saddle points* **between regions:** Left image shows the iso-action levels near the saddle point between $\mathbf{R}_1$ and $\mathbf{R}_2$

We use the fact that given two regions $\mathbf{R}_1$ and $\mathbf{R}_2$, the saddle point $\mathbf{s}$ where the two fronts starting from each region meet can be used to find the minimal path between $\mathbf{R}_1$ and $\mathbf{R}_2$. Indeed, the minimal path between the two regions has to pass by the meeting point $\mathbf{s}$. This point is the point half way (in energy) on a minimal path between $\mathbf{R}_1$ and $\mathbf{R}_2$. Back-propagating from $\mathbf{s}$ to $\mathbf{R}_1$ and then from $\mathbf{s}$ to $\mathbf{R}_2$ gives the two halves of the path.

### Notations and definitions

Here are some definitions that will be used in what follows. $X$ being a set of points in the image, $U_X$ is the minimal action obtained by Fast Marching with potential $\tilde{P}$ and starting points $\{p, p \in X\}$. This means that all points of $X$ are initialized as alive points with value 0. All their 4-connexity neighbors that are not in $X$ are *trial* points. This is easy to see that $U_X = \min_{p \in X} U_p$. $X$ may be a connected component $R$ or a set of connected components.

The *label* $l$ at a point $p$ is equal to the index $k$ of the region $\mathbf{R}_k$ for $p$ closer in energy to $\mathbf{R}_k$ than to other regions $\mathbf{R}_j$. This means that minimal action $U_{\mathbf{R}_k}(p) \leq U_{\mathbf{R}_j}(p), \forall j \neq k$. We define the region $L_k = \{p / l(p) = k\}$. If $X = \cup_j \mathbf{R}_j$, we have $U_X = U_{\mathbf{R}_k}$ on $L_k$ and the computation of $U_X$ is the same as the simultaneous computation of each $U_{\mathbf{R}_k}$ on each region $L_k$. These are the simultaneous fronts starting from each $\mathbf{R}_k$.

A *saddle point* $\mathbf{s}(\mathbf{R}_i, \mathbf{R}_j)$ between $\mathbf{R}_i$ and $\mathbf{R}_j$ is the first point where the front starting from $\mathbf{R}_i$ to compute $\mathcal{U}_{\mathbf{R}_i}$ meets the front starting from $\mathbf{R}_j$ to compute $\mathcal{U}_{\mathbf{R}_j}$; At this point, $U_{\mathbf{R}_i}$ and $U_{\mathbf{R}_j}$ are equal and this is the smallest value for which they are equal.

Two different regions among the $\mathbf{R}_k$'s will be called *linked regions* if they are selected to be linked together. The way we choose to link two regions is to select some *saddle points*. Thus regions $\mathbf{R}_i$ and $\mathbf{R}_j$ are *linked regions* if their *saddle point* is among the selected ones.

A *cycle* is a sequence of different regions $\mathbf{R}_k, 1 \leq k \leq K$, such that for $1 \leq k \leq K-1$, $\mathbf{R}_k$ and $\mathbf{R}_{k+1}$ are *linked regions* and $\mathbf{R}_K$ and $\mathbf{R}_1$ are also *linked regions*.

### Finding and Selecting Saddle Points

The main goal of our method is to obtain all significant paths joining the given regions. However, each region should not be connected to all other regions, but only to those that are closer to them in the energy sense. There are many possibilities for deciding which regions connect together depending on the kind of data and application. In some cases, the goal would be to detect closed curves and avoid forming branches, as in [31]. Then the criterion would be to constrain a region to be linked to at most two other regions in order to make *cycles*. In our context, we are interested in detecting branches and avoiding closed curves. Therefore the criterion for two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ to be connected is that their fronts meet without creating a "cycle".

We see in Figure 9.18 a zoom on the saddle points detected between regions $\mathbf{R}_1$ and $\mathbf{R}_2$ and $\mathbf{R}_3$ and $\mathbf{R}_4$. Once a *saddle point* $\mathbf{s}(\mathbf{R}_i, \mathbf{R}_j)$ is found and selected, back-propagation relatively to final energy $U$ should be done both ways to $\mathbf{R}_i$ and to $\mathbf{R}_j$ to find the two halves of the path between them. We see in Figure 9.19 this back-propagation at each of the three automatically selected *saddle points*. They link $\mathbf{R}_1$

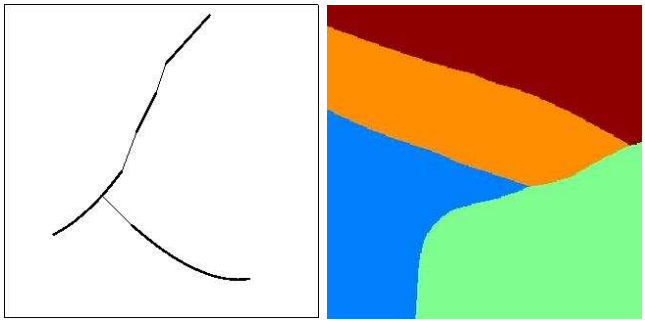


**Figure 9.19. Example with four regions:** On the left we show the minimal paths obtained by back-propagation from the three *saddle points* to each of the regions from where the front comes; on the right, and the Voronoi diagram obtained.

to $\mathbf{R}_2$, $\mathbf{R}_2$ to $\mathbf{R}_3$ and $\mathbf{R}_3$ to $\mathbf{R}_4$. At a saddle point, the gradient is zero, but the direction of descent towards each point are opposite. For each back-propagation, the direction of descent is the one relative to each region. This means that in order to estimate the gradient direction toward $\mathbf{R}_i$, all points in a region different from $L_i$ have their energy put artificially to $\infty$. This allows finding the good direction for the gradient descent towards $\mathbf{R}_i$. However, as mentioned earlier, these back-propagations have to be done only for selected *saddle points*. In the fast marching algorithm we have a simple way to find *saddle points* and update the *linked regions*.

As defined above, the region $L_k$ associated with a region $\mathbf{R}_k$ is the set of points $p$ of the image such that minimal energy $U_{\mathbf{R}_k}(p)$ to $\mathbf{R}_k$ is smaller than all the $\mathcal{U}_{\mathbf{R}_j}(p)$ to other regions $\mathbf{R}_j$. The set of such regions $L_k$ covers the whole image, and forms the Voronoi diagram of the image (see figure 9.19). All *saddle points* are at a boundary between two regions $L_k$. For a point $p$ on the boundary between $L_j$ and $L_k$, we have $\mathcal{U}_{\mathbf{R}_k}(p) = U_{\mathbf{R}_j}(p)$. The *saddle point* $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_j)$ is a point on this boundary with minimal value of $U_{\mathbf{R}_k}(p) = U_{\mathbf{R}_j}(p)$. This gives us a rule to find the *saddle points*

**Minimal paths between Regions $\mathbf{R}_k$**

- Initialization:
  - $\mathbf{R}_k$'s are given
  - $\forall k, \forall p \in \mathbf{R}_k, V(p) = 0; l(p) = k$; $p$ *alive*.
  - $\forall p \notin \cup_k \mathbf{R}_k, V(p) = \infty; l(p) = -1$; $p$ is *far* except 4-connexity neighbors of $\mathbf{R}_k$'s that are *trial* with estimate $U$ using equation (1.7).
- Loop for computing $V = U_{\cup_k \mathbf{R}_k}$:
  - Let $p = (i_{min}, j_{min})$ be the *Trial* point with the smallest action $U$;
  - Move it from the *Trial* to the *Alive* set with V(p) = U(p);
  - Update $l(p)$ with the same index as point $A_1$ in formula (1.7). If $R(A_1) \neq R(B_1)$ and we are in case 1 in table 2.2 where both points are used and if this is the first time regions of labels $l(A_1)$ and $l(B_1)$ meet, $\mathbf{s}(\mathbf{R}_{l(A_1)}, \mathbf{R}_{l(B_1)}) = p$ is set as a *saddle point* between $\mathbf{R}_{l(A_1)}$ and $\mathbf{R}_{l(B_1)}$. If adding a link between these regions does not create a *cycle*, they are set as *linked regions* and $\mathbf{s}(\mathbf{R}_{l(A_1)}, \mathbf{R}_{l(B_1)}) = p$ is selected, For each neighbor $(i, j)$ of $(i_{min}, j_{min})$:
    - If $(i, j)$ is *Far*, add it to the *Trial* set;
    - If $(i, j)$ is *Trial*, update action $U_{i,j}$.
- Obtain all paths between selected *linked regions* by back-propagation each way from their *saddle point* (see Section 9.3.1).

**Table 9.1.** Algorithm of Section 9.3.1

during the fast marching algorithm.

Each time two fronts coming from $\mathbf{R}_k$ and $\mathbf{R}_j$ meet for the first time, we define the meeting point as $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_j)$. This means that we need to know for each point of the image from where it comes. This is easy to keep track of its origin by generating an index map updated at each time a point is set as alive in the algorithm. Each point of region $\mathbf{R}_k$ starts with label $k$. Each time a point is set as alive, it gets the same label as the points it was computed from in formula (1.7). In that formula, the computation of $U_{i,j}$ depends only on at most two of the four pixels involved. These two pixels, said $A_1$ and $B_1$, have to be with the same *label*, except if $(i, j)$ is on the boundary between two labels. If $A_1$ and $B_1$ are both alive and with different labels $k$ and $l$, this means that regions $\mathbf{R}_k$ and $\mathbf{R}_l$ meet there. If this happens for the first time, the current point is set as the *saddle point* $\mathbf{s}(\mathbf{R}_k, \mathbf{R}_l)$ between these regions. A point on the boundary between $\mathbf{R}_k$ and $\mathbf{R}_l$ is given the label of the neighbor point with smaller action $A_1$. At the boundary between two labels there can be a slight error on labeling. This error of at most one pixel is not important in our context and could be refined if necessary.

### Algorithm

The algorithm for this section is described in Table 9.1 and illustrated in figures 9.17 to 9.19. When there is a large number of $\mathbf{R}_k$'s, this does not change much the computation time of the minimal action map, but this makes more complex dealing with the list of linked regions and *saddle points* and testing for *cycles*.

The way we chose to test for cycles is as follows. Assume a saddle point between regions $\mathbf{R}_i$ and $\mathbf{R}_j$ is found. We then test if there is already a link between these regions through other regions. This means we are looking for a sequence of different regions $\mathbf{R}_k, 1 \leq k \leq K$, with $\mathbf{R}_1 = \mathbf{R}_i$ and $\mathbf{R}_K = \mathbf{R}_j$, such that for $1 \leq k \leq K-1$, $\mathbf{R}_k$ and $\mathbf{R}_{k+1}$ are *linked regions*.

This kind of condition can be easily implemented using a recursive algorithm. When two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ are willing to be connected - i.e. that their fronts meet - a table storing the connectivity between each region enables to detect if a link already exists between those regions. Having $N$ different regions, we fill a matrix $M(N,N)$ with zeros, and each time two regions $\mathbf{R}_i$ and $\mathbf{R}_j$ meet without creating a cycle, we set $M(i,j) = M(j,i) = 1$. Thus, when two regions meet, we apply the algorithm detailed in table 9.2.

**Algorithm for Cycle detection** when a region $\mathbf{R}_i$ meets a region $\mathbf{R}_j$:
*Test*$(i,j,M,i)$; *with*
*Test*$(i,j,M,l)$;

- if $M(l,j) = 1$, return 1;
- else
  - count=0;
  - for $k \in [1,N]$ with $k \neq i, k \neq j, k \neq l$ : count $+=$ *Test*$(k,j,M,l)$;
  - return count;

**Table 9.2.** Cycle detection

If two regions are already linked, the pixel where their fronts meet is not considered as a valuable candidate for back-propagation. The algorithm stops automatically when all regions are connected.

### Application

The method can be applied to connected components from a whole set of edge points or points obtained through a preprocessing. Finding all paths from a given set of points is interesting in the case of a binary potential defined, like in Figure 9.17, for *Perceptual Grouping* . It can be used as well when a special preprocessing is possible, either on the image itself to extract characteristic points or on the geometry of the initial set of points to choose more relevant points. We show in figures 9.20 and 9.21 an example of application to a medical image of the hip where the objects of interest are the vessels. Potential $P$ is defined using ideas from [60] on vesselness filter (detailed later in section 9.3.2). About vessel detection, see also [107, 177].

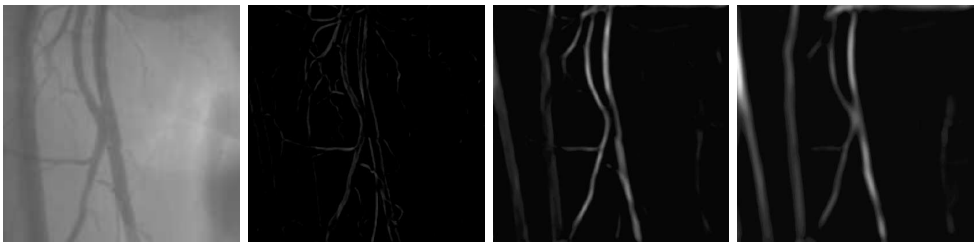


**Figure 9.20. Multiscale vessel enhancement**: First image is the original dataset; All other images are from left to right the filter response with respective kernels 1, 3, and 5.

**Figure 9.21. *Perceptual Grouping* on a 2D Medical Image**: left image is the vesselness potential; middle and right images show that from the set of regions obtained from thresholding of potential image, our method finds links between these regions as minimal paths with respect to the potential.

### 9.3.2 Finding a Set of Paths in a 3D Image

#### Extension to 3D

We now extend our approach to finding a set of 3D minimal paths between regions in 3D images. All definitions and algorithms of section 9.3.1 are not affected by changing the dimension of the image from 2D to 3D. The main changes are that 4-connexity in 2D is now 6-connexity in 3D and that we deal with minimal paths and minimal action in 3D images (see section 2.1 for the 3D extension of the the fast marching).

#### Application to Real Datasets: a MR Image of the Aorta

The problem here is to complete a partially detected object. In figure 3.12 is shown a 3D MR dataset of the aorta, which presents a typical pathology: an abdominal aortic aneurysm. The anatomical object is made visible on the image by injecting a contrast product before the image acquisition.

We propose here to give a method for extracting from the grey level image a set of paths that will represent an approximate skeleton of the tree structure. This is based on extracting first a set of unstructured voxels or regions that belong to the

object. Notice that [107,177] give different methods to detect vessels but ours is much simpler and faster.

For this, we propose to extract valuable information from this dataset, computing a multi-scale vessel enhancement measure, based on the work of [60] on ridge filters. Having extracted the three eigenvalues of the Hessian matrix computed at scale $\sigma$, ordered $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$, we define the vesselness function as done in the preceding chapter 8.

In figure 9.22 you can observe the response of the filter, based on the Hessian information, at three different scales: $\sigma = 1, 5, 10$. Visualization is made with Maximum Intensity Projection (MIP). Using this information computed at several scales,

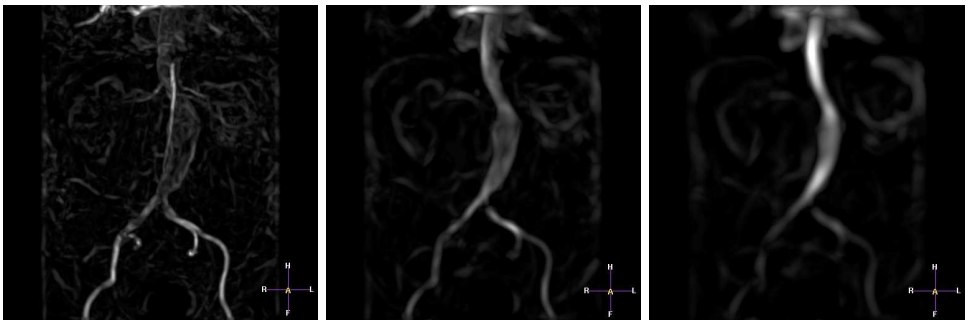


**Figure 9.22. Ridge detection at three different scales:** $\sigma = 1, 5, 10$ (MIP visualization of the 3D images)

we can take as potential the maximum of the response of the filter across all scales (Fig. 9.23-left). And we can easily give a very constrained threshold of this image, that will lead to sets of unstructured voxels that surely belong to the anatomical object of interest, as shown in figure 9.23-middle.

Based on this set of regions, we apply our algorithm of section 9.3.1, using the 3D version of the Fast-Marching algorithm presented in section 2.1. We find the set of paths that connect altogether all the seed regions in our image, leading to the representation shown in figure 9.23-right.

### 9.3.3 Conclusion

We presented a new method that finds a set of contour curves in an image. It was applied to *Perceptual Grouping* to get complete curves from a set of edge regions with gaps. The technique is based on finding minimal paths between two end points [34]. However, in our approach, start and end points are not required as initialization. Given a unstructured set of regions, the pairs of regions that had to be linked by minimal paths are automatically found. Once *saddle points* between pairs of regions are found, paths are drawn on the image from the selected *saddle points* to both points of each pair. This gives the minimal paths between selected pairs of regions. The whole set of paths completes the initial set of contours and allows to close these contours. We applied this method in order to reconstruct vascular structures, and we

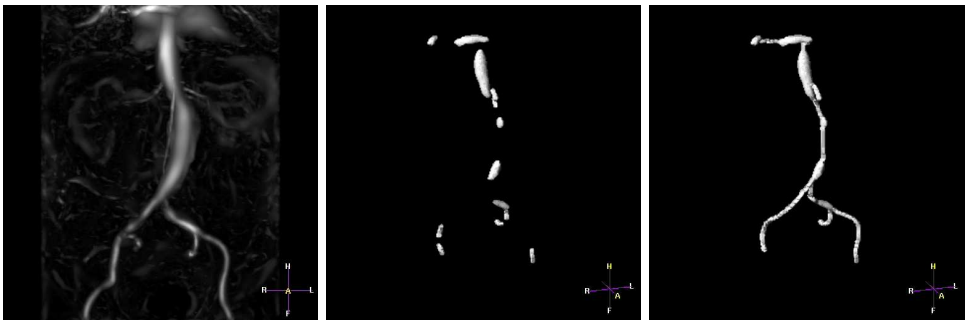


**Figure 9.23. *Perceptual Grouping* in the aorta of figure 3.12:** from left to right, visualization of the 3D potential (MIP view) obtained from the different scale of previous figure; a rough detection of the aorta; the Reconstructed aorta.

showed examples for 2D vascular image and 3D medical dataset of the aorta. In case a refinement is needed, this method could be an efficient way to initialize geodesic contours. Other developments could lead to applications in roads detection in aerial images [63].

# Conclusion

## Summary and contributions

In this manuscript, we focused on curves and shapes extraction in 3D medical imaging.

### Path Extraction

We developed a set of original methods, based on the work of *Cohen and Kimmel* [34], in order to extend the minimal path extraction to three dimensional datasets. We worked upon providing several algorithms in order to reduce the computational cost of those methods and to the ease the use of minimal path techniques by any kind of possible users, clinicians among others.

Those results lead to several applications, and among them the implementation of a system of automatic path extraction of trajectories for virtual endoscopy, which has been clinically validated, and is now integrated in a commercial product, *EasyVision*, a software package for image processing sold with acquisition systems, and developed by Philips Medical Systems.

Second application is the construction of a tool for interactive and real-time extraction of objects contours in 2D images, built upon the basis of models such as the *Live-Wire* of *Falcao, Udupa, Mortensen and Barrett* [49, 127]. Our goal was to provide a tool for semi-automatic contour extraction, using the minimal path extraction principles. The resulting method integrated an interesting facility: the user can teach the algorithm which kind of contours he is looking for. This method gives interesting and promising results, and will probably be integrated in a software package for image processing, for delineation of the heart ventricles in ultrasound images.

### Surface Extraction

The second part of the thesis focuses on the extraction of surfaces, with the help of minimal path extraction algorithms. We explained the link which exists with similar techniques in mathematical morphology, particularly with the *Watersheds* of *Vincent* [180], and we demonstrated the interest of such a method, which is fast and can produce a valuable initialization for more complex algorithms, with bigger computing times like *Level-Sets*. We presented a collaborative method which combines those different techniques in a single framework.

This framework was applied to several complex segmentation and visualization problems, in the sense that the topology of the object is difficult to recover. We first applied our method to extraction of cerebral aneurysms, which are inflations of the brain vessels that can grow and break, leading to a brain hemorrhage. Those aneurysms have a wide variety of shapes and the suitable model for segmentation should not have any *a priori* on the structure of the final object to be recovered. Second application, using the same principle, is segmentation and visualization of colon polyps. We detailed a method to discriminate areas of the colon to be observed for detecting polyps, based on the curvature of the surface segmented. This last method will be the basis of further developments for automatic detection of polyps at Philips Medical Systems.

### Tree Structure Extraction

Finally, in the last part of the thesis we focused on the adaptation of our algorithms to the particular case of tree anatomical structures, where path and shape extraction find an original domain of development. We first develop a technique of fast tree segmentation, with a simple initialization (setting a seed point), adapted to tubular structures, with no constraint on the topology of the final object. We further developed a way to obtain an accurate sub-voxel segmentation method, using the previous algorithm as initialization.

In order to provide a complete analysis of the tree-shaped objects, it is important to use adequate tools to navigate inside the tree hierarchy, and a way to label the different parts of the structure. We first extent the minimal path extraction technique to the automatic detection of a whole set of trajectories, then we found a method to reconstruct the skeleton of our tubular anatomical objects, on the basis of those trajectories, by detecting the branching points and the paths between them.

Those techniques were applied to the segmentation and the reconstruction of vascular and arterial trees, in 3D contrast-enhanced angiographic medical images, and extended to bronchial trees in multislice CT scanners. Comparing our results obtained with a panel of methods already proposed in this domain, we concluded on the interest and validity of our framework, which provides a sub-voxelic accuracy of our tubular objects, in interactive times. Using the information of tree hierarchy, we can localize the branching points and extract the interesting information of the section of our objects in the whole volume. Pathologies like aneurysms, or stenoses can be clearly measured.

## Problems encountered and perspectives

Our tree extraction technique has not been clinically validated for the moment. Results are promising, but the initialization technique can be improved: it cannot recover the smaller parts of our structures, even if it is a very fast method. A possible perspective would be to use the perceptual grouping methods of section 9.3 as a second process in order to recover those smaller parts.

The perspectives of this study are essentially to make a clinical validation, as it was

done for the virtual endoscopy tool. However, many of the applications presented in each part of the thesis gives a direction for promising developments, like visualization of the colon polyps and extraction of tree structures.

Moreover, the mathematical methods used for path and surface extraction that we developed all along the thesis can be used in a more general perspective than medical imaging, and can be applied to other industrial applications, such as aerial images.

# Bibliography


[1] D. Adalsteinsson, R. Kimmel, R. Malladi, J.A. Sethian, "Center for Pure and Applied Mathematics, Univ. of California, Berkeley, CA 94720", Tech. Rep., Fast Marching Methods for Computing Solutions to Static Hamilton-Jacobi Equations, 1996.

[2] D. Adalsteinsson & J.A. Sethian, "A Fast Level Set Method for Propagating Interfaces", *Journal of Computational Physics*, vol. 118, pp. 269–277, 1995.

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

[4] A.A. Amini, T.E. Weymouth, R.C. Jain, "Using Dynamic Programming for Solving Variational Problems in Vision", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 12, no. 9, pp. 855–867, September 1990.

[5] F. Angella, *Modeles Deformables et Systemes Particulaires - Application a l'Extraction de Structures Arborescentes en Analyse d'Images*, Ph.D. thesis, Université Bordeaux I, 2001.

[6] F. Angella, O. Lavialle, P Baylou, "A deformable and Expansible Tree for Structure Recovery", in *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, vol. 1, pp. 241–245, 1996.

[7] C. Baillard, P. Hellier, C. Barillot, "Segmentation of brain 3D MR images using level sets and dense registration", *Medical Image Analysis*, vol. 5, no. 3, pp. 185–194, September 2001.

[8] F. Barbaresco & B. Monnier, "Minimal Geodesics Bundles by Active Contours : Radar Application for Computation of Most Threatening Trajectories Areas and Corridors", in *Proceedings of European Signal and Image Processing Conference, EUSIPCO'00*, Tampere, Finland, sept 2000.

[9] E. Bardinet, *Constrained deformable models - Applications to cardiac imagery*, Ph.D. thesis, University of Paris IX-Dauphine, 1995.

[10] W. Barrett & E. Mortensen, "Interactive live-wire boundary extraction", *Medical Image Analysis*, vol. 1, no. 4, pp. 331–341, 1997.

[11] R. Bellman & R. Kalaba, *Dynamic Programming and modern control theory*, London mathematical society monographs, London, 1965.

[12] M. Bertalmio, G. Sapiro, G. Randall, "Region Tracking on Level-Sets Methods", in *Scale Space Theories in Computer Vision - Scale Space'99*, pp. 330–338, 1999.

[13] J. Bloomenthal, *Introduction to Implicit Surfaces*, Morgan Kaufmann Publishers, San Francisco, California, 1997.

[14] J. Bloomenthal & K. Shoemake, "Convolution surfaces", *Computer Graphics*, vol. 25, no. 4, pp. 251–256, July 1991.

[15] H. Blum, "A transformation for extracting new descriptors of shape", in *Models for the Perception of Speech and Visual Forms*, W. Wathen-Dunn (ed.), MIT Press, Amsterdam, pp. 362–380, 1967.

[16] G. Borgefors, "Distance Transformations in Arbitrary Dimensions", *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 27, no. 3, pp. 321–345, September 1984.

[17] I. Bricault, G. Ferretti, P. Cinquin, "Multi-level Strategy for Computer-Assisted Transbronchial Biopsy", in *Proceedings of the first International Conference on Medical Image Computing and Computer-Assisted Intervention, (MICCAI'98)*, 1998.

[18] D. Buthiau, E. Antoine, J.C. Piette, D. Nizri, P. Baldeyrou, D. Khayat, "Virtual tracheobronchial endoscopy: educational and diagnostic value", *Surg Radiol Anat*, vol. 18, no. 2, pp. 125–131, 1996.

[19] M.P. Cani & M. Desbrun, "Animation of Deformable Models Using Implicit Surfaces", *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 1, March 1997.

[20] J. Canny, "A Computational Approach to Edge Detection", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 8, no. 6, pp. 679–698, November 1986.

[21] V. Caselles, F. Catté, T. Coll, F. Dibos, "A Geometric Model for Active Contours", *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

[22] V. Caselles, R. Kimmel, G. Sapiro, "Geodesic active contours", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, IEEE Computer Society Press, Cambridge, USA, pp. 694–699, 1995.

[23] V. Caselles, R. Kimmel, G. Sapiro, "Geodesic active contours", *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61–79, 1997.

[24] A. Chakraborty, L.H. Staib, J.S. Duncan, "Deformable Boundary Finding in Medical Images by Integrating Gradient and Region Information", *IEEE Transactions on Medical Imaging*, vol. 15, no. 6, pp. 859–870, 1996.

[25] T.F. Chan & L.A. Vese, "Active contours without edges", *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 266–277, February 2001.

[26] S. Chandran, T. Meajima, S. Miyazaki, "Global minima via dynamic programming: Energy minimising active contours", in *Proceedings SPIE Geometric Methods in Computer Vision*, vol. 1570, pp. 391–402, 1991.

[27] C. Chesnaud, P. Refregier, V. Boulet, "Statistical Region Snake-Based Segmentation Adapted to Different Physical Noise Models", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 21, no. 11, pp. 1145–1157, November 1999.

[28] D.L. Chopp & J.A. Sethian, "Motion by Intrinsic Laplacian of Curvature", Tech. Rep., Dept. of Mathematics, Univ. of California, Berkeley, September 1998. CPAM Report PAM-746.

[29] L.D. Cohen, "On active contour models and balloons", *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 53, no. 2, pp. 211–218, 1991.

[30] L.D. Cohen, *Variational Methods for Image Processing*, Université Paris Dauphine, May 1995. "Mémoire d'Habilitation à diriger des recherches" Presented together with 10 main publications (in English) during 1988-1995.

[31] L.D. Cohen, "Multiple Contour Finding and Perceptual Grouping using Minimal Paths", *Journal of Mathematical Imaging and Vision*, vol. 14, no. 3, 2001. CEREMADE TR 0101, Jan 2001. To appear.

[32] L.D. Cohen & I. Cohen, "Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images", *PAMI*, vol. 15, no. 11, pp. 1131–1147, November 1993.

[33] L.D. Cohen & R. Kimmel, "Fast marching the global minimum of active contours", in *Proceedings of the IEEE International Conference on Image Processing (ICIP'96)*, vol. 1, Lausanne, Switzerland,, pp. 473–476, sept 1996.

[34] L.D. Cohen & R. Kimmel, "Global Minimum for Active Contour Models: A Minimal Path approach", *International Journal of Computer Vision*, vol. 24, no. 1, pp. 57–78, Aug. 1997.

[35] M.G. Crandall, H. Ishii, P. L. Lions, "User's guide to viscosity solutions of second order partial linear differential equations", *Bulletin of the American Math. Society*, vol. 27, pp. 1–67, 1992.

[36] O. Cuisenaire, *Distance Transformations: Fast Algorithm and Applications to medical Image Processing*, Ph.D. thesis, Université catholique de Louvain, Belgium, Oct. 1999.

[37] H. Delingette, *Modélisation, déformation et reconnaissance d'objets tridimensionnels à l'aide de maillages simplexes*, Ph.D. thesis, Ecole Centrale Paris, France, July 1994.

[38] H. Delingette, "General Object Reconstruction Based on Simplex Meshes", *International Journal of Computer Vision*, vol. 32, no. 2, pp. 1–36, September 1999.

[39] H. Delingette & J. Montagnat, "New Algorithms for Controlling Active Contours Shape and Topology", in *Proceedings of the Sixth European Conference on Computer Vision (ECCV'00)*, pp. xx–yy, 2000.

[40] H. Delingette & J. Montagnat, "Shape and Topology Constraints on Parametric Active Contours", *CVIU*, vol. 83, no. 2, pp. 140–171, August 2001.

[41] M. Desbrun, *Modeling and Animating highly deformable objects in Computer Graphics.*, Ph.D. thesis, Institut National Polytechnique de Grenoble, December 1997.

[42] T. Deschamps, S.M. Ebeid, L.D. Cohen, "Image Processing Method, System and Apparatus for Processing an Image representing a tubular structure and for constructing a path related to said structure", Patent Pending, March 1999.

[43] E.W. Dijkstra, "A note on two problems in connection with graphs", *Numerische Mathematic*, vol. 1, pp. 269–271, 1959.

[44] L.C. Evans & J. Spruck, "Motion of level-sets by mean curvature", *Journal of Differential Geometry*, vol. 33, pp. 635–681, 1991.

[45] R. Fahrig, A.J. Fox, S. Lownie, D.W. Holdsworth, "Use of a C-Arm System to Generate True Three-dimensional Computed Rotational Angiograms: Preliminary In Vitro and In Vivo Results", *American Journal Neuroradiology*, vol. 18, pp. 1507–1514, 1997.

[46] A.X. Falcao & J.K. Udupa, "Segmentation of 3D objects using live-wire", in *SPIE Medical Imaging*, vol. 3034, Newport Beach, CA, pp. 228–239, February 1997.

[47] A.X. Falcao & J.K. Udupa, "A 3D generalization of user-steered live-wire segmentation", *Medical Image Analysis*, vol. 4, no. 4, pp. 389–402, 2000.

[48] A.X. Falcao, J.K. Udupa, S. Samarasekera, B.E. Hirsch, "User-steered image boundary segmentation", in *SPIE Proceedings*, vol. 2710, pp. 278–288, 1996.

[49] A.X. Falcao, J.K. Udupa, S. Samarasekera, S. Sharma, B.E. Hirsch, R. de A. Lotufo, "User-Steered Image Segmentation Paradigms: Live-Wire and Live-Lane", *Graphical Models and Image Processing*, vol. 60, no. 4, pp. 233–260, 1998.

[50] O.D. Faugeras & R. Keriven, "Variational-Principles, Surface Evolution, PDEs, Level Set Methods, and the Stereo Problem", *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 336–344, March 1998.

[51] L.A. Feldkamp, L.C. Davis, J.W. Kress, "Practical cone-beam algorithm", *Journal of the Optical Society of America*, vol. 1, no. 6, pp. 612–619, June 1984.

[52] C. Fetita, *Analyse morphofonctionnelle des voies aériennes en TDM spiralée volumique*, Ph.D. thesis, Université Paris 5 René Descartes, 2000.

[53] M.A. Fischler, J.M. Tenenbaum, H.C. Wolf, "Detection of roads and linear structures in low-resolution aerial imagery using a multi-source knowledge integration technique", *Computer Graphics and Image Processing*, vol. 15, pp. 201–223, 1981.

[54] R. Florent, "Image processing method and system, and medical examination apparatus for extracting a path following a threadlike structure in an image", Patent Pending, June 1999, 99401348.0.

[55] R. Florent & J. Breitenstein, "Guide-wire extraction in fluoroscopy", Patent Pending, December 1998, 98403323.3.

[56] R. Florent & L. Goubet, "Extraction of Guide-wire with ridgeness + front propagation", Patent Pending, March 2000, 00 401367.8.

[57] A. Frangi, *Three-Dimensional Model-Based Analysis of Vascular and Cardiac images*, Ph.D. thesis, University Medical Center Utrecht, 2001.

[58] A. Frangi, W. Niessen, R.M. Hoogeveen, Th. van Walsum, M.A. Viergever, "Model based quantification of 3D resonance magnetic angiographic images", *IEEE Transactions on Medical Imaging*, vol. 18, no. 10, pp. 946–956, October 1999. Special Issue on Model-based Medical Image Analysis.

[59] A. Frangi, W. Niessen, P.J. Nederkoorn, J. Bakker, W.P.Th.M. Mali, M.A. Viergever, "Quantitative analysis of vessel morphology from 3D MR angiograms:in vitro and in vivo results", *Magnetic Resonance in Medicine*, vol. 45, no. 2, pp. 311–322, February 2001.

[60] A. Frangi, W. Niessen, K.L. Vincken, M.A. Viergever, "Multiscale Vessel Enhancement Filtering", in *Proceedings of the first International Conference on Medical Image Computing and Computer-Assisted Intervention, (MICCAI'98)*, pp. 130–137, 1998.

[61] P. Fua & Y.G. Leclerc, "Model driven edge detection", *Machine Vision and Applications*, vol. 3, pp. 45–56, 1990.

[62] D. Geiger, A. Gupta, L. Costa, J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 17, no. 3, March 1995.

[63] D. Geman & B. Jedynak, "An Active Testing Model for Tracking Roads in Satellite Images", *PAMI*, vol. 18, no. 1, pp. 1–14, January 1996.

[64] J. Gil & R. Kimmel, "Efficient Dilation, Erosion, Opening and Closing Algorithms", in *International Symposium on Mathematical Morphology and its Applications to Image and Signal Processing V*, Palo-Alto, CA, USA, June 2000.

[65] R. Goldenberg, R. Kimmel, E. Rivlin, M. Rudzsky, "Fast Geodesic Active Contours", in *Scale Space Theories in Computer Vision - Scale Space'99*, pp. 34–45, 1999.

[66] R. Goldenberg, R. Kimmel, E. Rivlin, M. Rudzsky, "Cortex Segmentation: A fast Variational Geometric Approach", in *Workshop on Variational and Level Set Methods in Computer Vision*, Vancouver, Canada, 2001.

[67] R. Goldenberg, R. Kimmel, E. Rivlin, M. Rudzsky, "Fast Geodesic Active Contours", *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1467–1475, October 2001.

[68] J. Gomes & O.D. Faugeras, "Reconciling Distance Functions and Level Sets", in *Scale Space Theories in Computer Vision - Scale Space'99*, pp. 70–81, 1999.

[69] J. Gomes & O.D. Faugeras, "Level Sets and Distance Functions", in *Proceedings of the Sixth European Conference on Computer Vision (ECCV'00)*, 2000.

[70] M. Grayson, "The Heat equation shrinks embedded plane curves to round points", *Journal of Differential Geometry*, vol. 26, pp. 285–314, 1987.

[71] M. Greff, O. Gerard, T. Deschamps, "Adaptation of potential terms in real-time optimal path extraction", Patent Pending, April 2001, FR 01 401 696.8.

[72] G. Guy & G. Medioni, "Inferring Global perceptual contours from local features", *International Journal of Computer Vision*, vol. 20, no. 1/2, pp. 113–133, Oct. 1996.

[73] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, "Nondistorting flatening maps and the 3D visualization of colon CT images", *IEEE Transactions on Medical Imaging*, July 2000.

[74] S. Haker, A. Tannenbaum, R. Kikinis, "Mass preserving mappings and image registration", in *Proceedings of the 4th International Conference on Medical Image Computing and Computer-Assisted Intervention, (MICCAI'01)*, W. Niessen & M.A. Viergever (eds.), Springer Verlag, Utrecht, The Netherlands, pp. 120–127, October 2001.

[75] J. Helferty, A. Sherbondy, A. Kiraly, J. Turlington, E.A. Hoffman, G. McLennan, W. Higgins, "Image-guided Endoscopy for Lung-cancer Assessment", in *Proceedings of the IEEE International Conference on Image Processing (ICIP'01)*, pp. 307–310, 2001.

[76] G. Hermosillo, O. Faugeras, J. Gomes, "Unfolding the Cerebral Cortex Using Level Set Methods", in *Scale Space Theories in Computer Vision - Scale Space'99*, pp. 58–69, 1999.

[77] L. Hong, S. Muraki, A. Kaufman, D. Bartz, H. Taosong, "Virtual voyage: interactive navigation in the human colon", in *Proceedings of 24th International Conference on Computer Graphics and Interactive Techniques*, pp. 27–34, August 1997.

[78] S. Jehan-Besson, M. Barlaud, G. Aubert, "Detection and Tracking of Moving Objects Using a New Level Set Based Method", in *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'00)*, pp. Vol III: 1112–1117, 2000.

[79] S. Jehan-Besson, M. Barlaud, G. Aubert, "Video Objects Segmentation Using Eulerian Region-Based Active Contours", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'01)*, vol. 1, pp. 353–360, 2001.

[80] F.A. Jolesz, W.E. Loresen, H. Shinmoto, H. Atsumi, S. Nakajima, P. Kavanaugh, P. Saiviroonporn, S.E. Seltzer, S.G. Silverman, M. Phillips, R. Kikinis, "Interactive virtual endoscopy", *American Journal of Radiology*, vol. 169, pp. 1229–1237, 1997.

[81] G.K. Kanizsa, "Subjective Contours", in *Scientific American*, no. 234, 1976.

[82] M. Kass, A. Witkin, D. Terzopoulos, "Snakes: Active contour models", *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[83] R. Kemkers, J. Op de Beek, H. Aerts, R. Koppe, E. Klotz, J.M. Moret, "3D-Rotational Angiography: First clinical application with use of a standard Philips C-arm system", in *Computer Assisted Radiology and Surgery, CARS'98*, 1998.

[84] R. Keriven, *Equations aux Dérivées Partielles, Evolutions de Courbes et de Surfaces et Espaces d'Echelle: Application à la Vision par Ordinateur*, Ph.D. thesis, Ecole Nationale des Ponts et Chaussées, 1997.

[85] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, A. Yezzi, "Gradient flows and geometric active contour models", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, IEEE Computer Society Press, Cambridge, USA, pp. 810–815, 1995.

[86] R. Kimmel, A. Amir, A.M. Bruckstein, "Finding shortest paths on surfaces using level sets propagation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 1, pp. 635–640, 1995.

[87] R. Kimmel, N. Kiryati, A.M. Bruckstein, "Distance maps and weighted distance transforms", *Journal of Mathematical Imaging and Vision*, vol. 6, pp. 223–233, May 1996. Special Issue on Topology and Geometry in Computer Vision.

[88] R. Kimmel, N. Kiryati, A.M. Bruckstein, "Multivalued Distance Maps for Motion Planning on Surfaces with Moving Obstacles", *IEEE Trans. Robotics and Automation*, vol. 14, no. 3, pp. 427–436, June 1998.

[89] R. Kimmel & J.A. Sethian, "Fast Marching Methods for Computing Distance Maps and Shortest Paths, CPAM Report 669", Tech. Rep., Center for Pure and Apllied Mathematics, University of California, Berkeley, 1996.

[90] R. Kimmel & J.A. Sethian, "Computing Geodesic Paths on Manifolds", in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 15, University of California, Berkeley, pp. 8431–8435, July 1998.

[91] R. Kimmel & J.A. Sethian, "Optimal algorithm for shape from shading and path planning", in *Mathematical Image Analysis, (MIA'00)*, Paris, September 2000.

[92] R. Kimmel & J.A. Sethian, "Optimal algorithm for shape from shading and path planning", *Journal of Mathematical Imaging and Vision*, vol. 14, no. 3, pp. 237–244, 2001.

[93] N. Kiryati & G. Szekely, "Estimating Shortest Paths and Minimal Distances on Digitized Three-Dimensional Surfaces", *Pattern Recognition*, vol. 26, pp. 1623–1637, 1993.

[94] J.J Koenderink & A.J. van Doorn, "Surface Shape and Curvature Scales", *Image and Vision Computing*, vol. 10, pp. 557–565, 1992.

[95] R. Koppe, E. Klotz, J. Op de Beek, H. Aerts, "3D Vessel reconstruction based in Rotational Angiography", in *Computer Assisted Radiology, CAR'95*, 1995.

[96] R. Kutka & S. Stier, "Extraction of Line Properties Based on Direction Fields", *IEEE Transactions on Medical Imaging*, vol. 15, no. 1, pp. 51–58, 1996.

[97] J.O. Lachaud, *Surface extraction from three-dimensional images: discrete approach and deformable model approach*, Ph.D. thesis, Université Joseph Fourier, july 1998.

[98] J.O. Lachaud & A. Montanvert, "Deformable meshes with automated topology changes for coarse-to-fine three-dimensional surface extraction", *Medical Image Analysis*, vol. 3, no. 2, pp. 187–207, 1999.

[99] S.H. Landis, T. Murray, S. Bolden, P.A. Wingo, "Cancer statistics, 1998", *Cancer Journal for Clinicians*, vol. 48, no. 1, pp. 6–29, 1998.

[100] J.C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, MA, 1991.

[101] O. Lavialle, F. Angella, P. Baylou, "Extension of the Minimal Path Searching for Structure Recovery", in *Proceedings of the IEEE International Conference on Image Processing (ICIP'99)*, Kobe, Japan, pp. 405–406, October 1999.

[102] F. Leymarie & M.D. Levine, "Tracking deformable objects in the plane using an active contour model", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 15, no. 6, pp. 617–634, 1993.

[103] S. Liapis, E. Sifakis, G. Tziritas, "Color And/or Texture Segmentation Using Deterministic Relaxation and Fast Marching Algorithms", in *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'00)*, vol. 3, pp. 621–624, 2000.

[104] W.E. Lorensen & H.E. Cline, "Marching Cubes: a high resolution 3D surface reconstruction algorithm", *Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[105] C. Lorenz, I.C. Carlsen, T.M. Buzug, C. Fassnacht, J. Weese, "A Multi-Scale Line Filter with Automatic Scale Selection Based on the Hessian Matrix for Medical Image Segmentation", in *Scale Space Theories in Computer Vision - Scale Space'97*, pp. 152–163, july 1997.

[106] L.M. Lorigo, O.D. Faugeras, W.E.L. Grimson, R. Keriven, R. Kikinis, A. Nabavi, C.F. Westin, "Codimension-Two Geodesic Active Contours for the Segmentation of Tubular Structures", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'00)*, vol. 1, pp. 444–451, 2000.

[107] L.M. Lorigo, O.D. Faugeras, W.E.L. Grimson, R. Keriven, R. Kikinis, A. Nabavi, C.F. Westin, "CURVES: Curve evolution for vessel segmentation", *Medical Image Analysis*, vol. 5, no. 3, pp. 195–206, September 2001.

[108] L.M. Lorigo, O.D. Faugeras, W.E.L. Grimson, R. Keriven, R. Kikinis, C.F. Westin, "Codimension 2 Geodesic Active Contours for MRA Segmentation", in *International Conference on Information Processing in Medical Imaging (IPMI'99)*, Visegrad, Hungary, June 1999.

[109] J.B.A. Maintz, P.A. van den Elsen, M.A. Viergever, "Evaluation of Ridge Seeking Operators for Multimodality Medical Image Matching", *PAMI*, vol. 18, no. 4, pp. 353–365, April 1996.

[110] R. Malladi, R. Kimmel, D. Adalsteinsson, G. Sapiro, V. Caselles, J.A. Sethian, "A Geometric Approach to Segmentation and Analysis of 3D Medical Images", in *Proceedings of mathematical methods in biomedical image analysis, MMBIA'96*, 1996.

[111] R. Malladi & J.A. Sethian, "A Real-Time Algorithm for Medical Shape Recovery", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'98)*, pp. 304–310, Jan. 1998.

[112] R. Malladi, J.A. Sethian, B.C. Vemuri, "Evolutionary Fronts for Topology-Independent Shape Modeling and Recovery", in *Proceedings of the Third European Conference on Computer Vision (ECCV'94)*, vol. A, Stockholm, Sweden, pp. 3–13, 1994.

[113] R. Malladi, J.A. Sethian, B.C. Vemuri, "Shape Modelling with Front Propagation: A Level Set Approach", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 17, no. 2, pp. 158–175, Feb. 1995.

[114] T. McInerney & D. Terzopoulos, "Topologically Adaptable Snakes", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, IEEE Computer Society Press, Cambridge, pp. 840–845, June 1995.

[115] T. McInerney & D. Terzopoulos, "Deformable Models in Medical Image Analysis: A Survey", *Medical Image Analysis*, vol. 1, no. 2, 1996.

[116] T. McInerney & D. Terzopoulos, "T-snakes: Topology adaptive snakes", *Medical Image Analysis*, vol. 4, no. 2, pp. 73–91, 2000.

[117] N. Merlet, J. Zerubia, K.A. Hogda, B. Braathen, K. Heia, "A curvature-dependent energy function for detecting lines in satellite images", in *Proceedings of Scandinavian Conference on Image Analysis*, Tromso, Norway, pp. 699–706, May 1993.

[118] F. Meyer & P. Maragos, "Multiscale Morphological Segmentations based on Watershed, Flooding, and Eikonal PDE", in *Scale Space Theories in Computer Vision - Scale Space'99*, pp. 351–362, 1999.

[119] J.S.B. Mitchell, D.W. Payton, D.M. Keirsey, "Planning and Reasoning for Autonomous Vehicle Control", *International Journal of Intelligent Systems*, vol. 2, pp. 129–198, 1987.

[120] J. Montagnat, *Modéles Déformables pour la Segmentation et la Modélisation d'Images Médicales 3D et 4D*, Ph.D. thesis, Université de Nice Sophia-Antipolis, France, December 1999.

[121] J. Montagnat & H. Delingette, "Globally constrained deformable models for 3D object reconstruction", *IEEE Transactions on Signal Processing*, vol. 71, no. 2, pp. 173–186, December 1998.

[122] U. Montanari, "On the Optimal Detection of Curves in Noisy Pictures", *Communications of the ACM*, vol. 14, no. 5, pp. 335–345, May 1971.

[123] K. Mori, J. Hasegawa, Y. Suenaga, J. Toriwaki, "Automated anatomical labeling of the bronchial branch and its application to the virtual bronchoscopy system", *IEEE Transactions on Medical Imaging*, vol. 19, no. 2, pp. 103–114, February 2000.

[124] K. Mori, J. Hasegawa, J. Toriwaki, H. Anno, K. Katada, "Recognition of Bronchus in Three Dimensional X-Ray CT Images with Applications to Virtualized Bronchoscopy System", in *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'96)*, pp. 528–532, 1996.

[125] K. Mori, S. Yamazaki, J. Hasegawa, "Extension of virtual bronchoscopy system as a teaching tool", in *Computer Assisted Radiology and Surgery, CARS'99*, pp. 166–170, 1999.

[126] E.N. Mortensen & W.A. Barrett, "Intelligent Scissors for Image Composition", in *Proceedings of Computer Graphics, SIGGRAPH'95*, pp. 191–198, 1995.

[127] E.N. Mortensen & W.A. Barrett, "Interactive Segmentation with Intelligent Scissors", *Graphical Models and Image Processing*, vol. 60, no. 5, pp. 349–384, September 1998.

[128] E.N. Mortensen & W.A. Barrett, "Toboggan-Based Intelligent Scissors with a Four Parameter Edge Model", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, pp. II:452–458, 1999.

[129] E. Mortensen, B. Morse, W. Barrett, J.K. Udupa, "Adaptive boundary detection using livewire two-dimensional dynamic programming", in *IEEE Proceedings of Computers in Cardiology*, pp. 635–638, October 1992.

[130] M. Naf, O. Kubler, R. Kikinis, M.E. Shenton, G. Szekely, "Characterization and Recognition of 3D Organ Shape in Medical Image Analysis Using Skeletonization", in *Proceedings of mathematical methods in biomedical image analysis, MMBIA'96*, pp. 139–150, 1996.

[131] D. Nain, S. Haker, R. Kikinis, W.E.L. Grimson, "An Interactive Virtual Endoscopy Tool", in *Workshop on Interactive Medical Image Visualization and Analysis*, W. Niessen, S. Olabarriaga, F. Gerritsen (eds.), Utrecht, The Netherlands, pp. 55–60, October 2001.

[132] T. O'Donnell, X.S. Fang, A. Gupta, T.E. Boult, "The Extruded Generalized Cylinder: A Deformable Model for Object Recovery", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pp. 174–181, 1994.

[133] S.D. Olabarriaga, *Human-Computer Interaction for the Segmentation of Medical Images*, Ph.D. thesis, University Medical Center Utrecht, December 1999.

[134] S.D. Olabarriaga & A.W.M. Smeulders, "Interaction in the segmentation of medical images: A survey", *Medical Image Analysis*, vol. 5, no. 2, pp. 127–142, June 2001.

[135] S. Osher & J.A. Sethian, "Fronts propagating with curvature dependent speed: algorithms based on the Hamilton-Jacobi formulation", *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[136] D.S. Paik, C.F. Beaulieu, R.B. Jeffrey, G.D. Rubin, S. Napel, "Automated flight path planning for virtual endoscopy", *Medical Physics*, vol. 25, no. 5, pp. 629–637, 1998.

[137] N. Paragios, *Geodesic Active Regions and Level Set Methods: Contributions and Applications in Artificial Vision*, Ph.D. thesis, Université de Nice Sophia-Antipolis, France, 2000.

[138] N. Paragios & R. Deriche, "Geodesic Active Regions for Motion Estimation and Tracking", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'99)*, Corfou, Greece, pp. 688–694, September 1999.

[139] N. Paragios & R. Deriche, "Geodesic Active Regions for Supervised Texture Segmentation", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'99)*, Corfou, Greece, September 1999.

[140] N. Paragios, O. Mellina-Gotardo, V. Ramesh, "Gradient Vector Flow Fast Geodesic Active Contours", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'01)*, vol. 1, Vancouver, Canada, pp. 67–73, 2001.

[141] G. Picinbono, "Modèle géométrique de contour déformable implicite pour la segmentation et la simulation", Master's thesis, Université de Nice-Sophia Antipolis, 1997.

[142] L. Piegl & W. Tiller, *The NURBS Book*, Springer Verlag, Berlin, 1996.

[143] C. Pisupati, L. Wolff, W. Mitzner, E. Zerhouni, "A Central Axis Algorithm for 3D Bronchial Tree Structures", in *SCV95*, pp. 259–264, 1995.

[144] F. Preteux, C. Fetita, P. Grenier, A. Capderou, "Modeling, segmentation and caliber estimation of bronchi in high-resolution computerized tomogrpahy", *Journal of Electronic Imaging*, vol. 8, no. 1, pp. 36–45, 1999.

[145] L.R. Rabiner, A.E. Rosenberg, S.E. Levinson, "Considerations In Dynamic Time Warping Algorithms For Discrete Word Recognition", *IEEE Transactions On Acoustics, Speech and Signal Processing*, vol. 26, no. 6, pp. 575–582, December 1978.

[146] P. Rogalla, J.T. van Schletinga, B. Hamm (eds.), *Virtual Endoscopy and Related 3D Techniques*, Diagnostic Imaging, Medical Radiology, Springer Verlag, Berlin, 2001.

[147] P. Rogalla, B. Verdonck, R. Truyen, B. Hamm, "Efficacy of automatic path tracking for virtual colonoscopy", in *Radiological Society of North America (RSNA)*, 1999.

[148] R. Ronfard, "Region-Based Strategies for Active Contour Models", *International Journal of Computer Vision*, vol. 13, no. 2, pp. 229–251, October 1994.

[149] N. Rougon & F. Preteux, "Generalized Geodesic Active Contours", in *Reconnaissance de Formes et Intelligence Artificielle (RFIA'98)*, Clermond-Ferrant, pp. 287–296, January 1998.

[150] E. Rouy & A. Tourin, "A Viscosity Solution Approach to Shape-From-Shading", *SIAM Journal of Numerical Analysis*, vol. 29, pp. 867–884, 1992.

[151] D. Rutovitz, "Data structures for operations on digital images", *Pictorial pattern recognition*, pp. 105–133, 1968.

[152] C. Samson, L. Blanc-Feraud, G. Aubert, J. Zerubia, "A Variational Model for Image Classification and Restoration", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 22, no. 5, pp. 460–472, May 2000.

[153] G. Sapiro, "Vector-valued active contours", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, San Francisco, USA, pp. 650–655, 1996.

[154] A. Sarti, R. Malladi, J.A. Sethian, "Subjective surfaces: A method for completing missing boundaries", in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 12, pp. 6258–6263, 2000.

[155] R. Sedgewick, *Algorithms in C - Parts 1-4*, 3 ed., Addison-Wesley, Reading, Massachusetts, USA, 1998.

[156] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, New York, 1982.

[157] J. Serra, "Morphological Descriptions Using Three-dimensional Wavefronts", Tech. Rep. [N -30/01/MM], Centre de Morphologie Mathématique, Ecole des Mines de Paris, June 2001.

[158] J.A. Sethian, "A review of recent numerical algorithms for hypersurfaces moving with curvature dependent flows", *Journal of Differential Geometry*, vol. 31, pp. 131–161, 1989.

[159] J.A. Sethian, "Curvature flow and entropy conditions applied to grid generation", *Journal of Computational Physics*, vol. 115, pp. 440–454, 1994.

[160] J.A. Sethian, "A review of the theory, algorithms, and applications of level set methods for propagating interfaces", *Acta Numerica*, 1995.

[161] J.A. Sethian, "A fast marching level set method for monotonically advancing fronts", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 93, no. 4, pp. 1591–1595, Feb. 1996.

[162] J.A. Sethian, "Tracking Interfaces with Level Sets", *American Scientist*, May-June 1997.

[163] J.A. Sethian, *Level set methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*, 2nd ed., Cambridge University Press, University of California, Berkeley, 1999.

[164] A. Shaashua & S. Ullman, "Structural saliency: The detection of globally salient structures using a locally connected network", in *Proc. Second IEEE International Conference on Computer Vision (ICCV'88)*, pp. 321–327, December 1988.

[165] J. Shah, "A common framework for curve evolution, segmentation and anisotropic diffusion", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, San Francisco, USA, 1996.

[166] P. Shirley & A. Tuchman, "A polygonal approximation to direct scalar volume rendering", *Computer Graphics*, vol. 24, no. 5, pp. 63–70, 1990.

[167] E. Sifakis, C. Garcia, G. Tziritas, "Bayesian level sets for image segmentation", in *Visual Communication and Image Representation*, 2001.

[168] American Cancer Society (ed.), *Cancer Facts and Figures*, 2, ACS, Atlanta, 1999.

[169] M. Sonka, W. Park, E. Hoffman, "Rule-Based Detection of Intrathoracic airway trees", *IEEE Transactions on Medical Imaging*, vol. 15, no. 3, pp. 314–326, June 1996.

[170] M. Sussman, P. Smereka, S.J. Osher, "A level set method for computing solutions to incompressible two-phase flow", *Journal of Computational Physics*, vol. 114, pp. 146–159, 1994.

[171] R. Szeliski, D. Tonnesen, D. Terzopoulos, "Modeling Surfaces of Arbitrary Topology with Dynamic Particles", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pp. 82–87, 1993.

[172] H. Tek & B. Kimia, "Image Segmentation by Reaction-Diffusion Bubbles", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, Cambridge, USA, pp. 156–162, June 1995.

[173] H. Tek & B. Kimia, "Boundary smoothing via symmetry transforms", *to appear in Journal of Mathematical Imaging and Vision, Special issue on Mathematics and Image Analysis MIA'00*, 2001.

[174] D. Terzopoulos, "On matching deformable models to images: Direct and iterative solutions", *Topical meeting on machine vision, Technical Digest Series, Optical Society of America*, vol. 12, pp. 160–167, 1987.

[175] D. Terzopoulos, A.P. Witkin, M. Kass, "Symmetry-Seeking Models and 3D Object Reconstruction", *International Journal of Computer Vision*, vol. 1, no. 3, pp. 211–221, October 1987.

[176] E. Thiel & A. Montanvert, "Chamfer masks: discrete distance functions, geometrical properties and optimization", in *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'92)*, The Hague, Netherlands, pp. 244–247, 1992.

[177] A. Vasilevskiy & K. Siddiqi, "Flux Maximizing Geometric Flows", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'01)*, vol. 1, Vancouver, Canada, pp. 149–154, 2001.

[178] P.W. Verbeek & B.J.H. Verwer, "Shading from shape, the eikonal equation solved by gray-weighted distance transform", *Pattern Recognition Letters*, vol. 11, pp. 681–690, 1990.

[179] A. Verroust & F. Lazarus, "Extracting skeletal curves from 3D scattered data", Tech. Rep., Unité de Recherche INRIA Rocquencourt, 1997. RR 3250.

[180] L. Vincent, *Algorithmes Morphologiques à base de Files d'Attente et de Lacets - Extension aux graphes*, Ph.D. thesis, Ecole des Mines de Paris, Xerox Imaging, Palo Alto, 1991.

[181] L. Vincent & P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 13, no. 6, pp. 583–598, June 1991.

[182] J. Weickert, B.M. ter Haar Romeny, M.A. Viergever, "Efficient and Reliable Schemes for Nonlinear Diffusion Filtering", *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 398–410, March 1998.

[183] R. Whitaker, "Algorithms for implicit deformable models", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, IEEE Computer Society Press, Cambridge, USA, pp. 822–827, 1995.

[184] R.T. Whitaker, "A Level-Set Approach to 3D Reconstruction from Range Data", *International Journal of Computer Vision*, vol. 29, no. 3, pp. 203–231, 1998.

[185] R.T. Whitaker & D.E. Breen, "Level-Set Models for the Deformation of Solid Objects", in *Proceedings of Implicit Surfaces, Eurographics/Siggraph (IS'98)*, pp. 19–35, June 1998.

[186] S. Wildermuth & C.H. Stern, "Interactive Definition of Endoluminal Aortic Stent Size and Morphology Based on Virtual Angioscopic Rendering of 3D MRA", in *Radiological Society of North America (RSNA)*, p. 226, 1998.

[187] L.R. Williams & D.W. Jacobs, "stochastic completion fields: a neural model of illusory contour shape and salience", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV'95)*, Cambridge, USA, pp. 408–415, June 1995.

[188] A. Witkin & P. Heckbert, "Using particles to sample and control implicit surfaces", in *Proceedings of the International Conference on Computer Graphics (SIGGRAPH'94)*, pp. 269–278, 1994.

[189] C.Y. Xu & J.L. Prince, "Gradient Vector Flow: A New External Force for Snakes", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*, pp. 66–71, 1997.

[190] C.Y. Xu & J.L. Prince, "Generalized gradient vector flow external forces for active contours", *IEEE Transactions on Signal Processing*, vol. 71, no. 2, pp. 131–139, December 1998.

[191] C.Y. Xu & J.L. Prince, "Snakes, Shapes, and Gradient Vector Flow", *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359–369, March 1998.

[192] G. Yeorong, D.R. Stelts, W. Jie, D.J. Vining, "Computing the centerline of a colon: a robust and efficient method based on 3D skeletons", *Journal of Computer-Assisted Tomography*, vol. 23, no. 5, pp. 786–794, 1999.

[193] A. Yezzi, A. Tsai, A. Willsky, "Medical image segmentation via coupled curve evolution equations with global constraints", in *Proceedings of mathematical methods in biomedical image analysis, MMBIA'00*, Hilton Head Island, USA, pp. 12–19, June 2000.

[194] X. Zeng, L.H. Staib, R.T. Schultz, J.S. Duncan, "Volumetric Layer Segmentation Using Coupled Surfaces Propagation", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'98)*, pp. 708–715, 1998.

[195] X. Zeng, L.H. Staib, R.T. Schultz, J.S. Duncan, "Segmentation and Measurement of the Cortex from 3-D MR Images Using Coupled-Surfaces Propogation", *IEEE Transactions on Medical Imaging*, vol. 18, no. 10, pp. 927–937, October 1999.

[196] S.C. Zhu & A. Yuille, "Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation", *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 18, no. 9, pp. 884–900, September 1996.

# Publications

## Publications in International Journals:

- T. Deschamps, L. D. Cohen, "Fast extraction of minimal paths in 3D images and applications to virtual endoscopy", *Medical Image Analysis*, Vol. 5, No. 4 , August 2001.

## Publications in International Conference Proceedings:

- T. Deschamps and L. D. Cohen, "Grouping connected components using minimal path techniques", *Computer Vision and Pattern Recognition (CVPR'01)*, Kauai, Hawai, December 2001
- R. Truyen, T. Deschamps and L. D. Cohen, "Clinical evaluation of an automatic path tracker for virtual colonoscopy." *Medical Image Computing and Computer-Assisted Intervention (MICCAI'01)*, Utrecht, Netherlands, October 2001.
- T. Deschamps, L. D. Cohen, "Multiple path extraction and perceptual grouping as a set of energy minimizing paths." *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR'01)*, Sophia-Antipolis, France, September 2001.
- R. Truyen, P. Lefere, S. Gryspeerdt and T. Deschamps, "Efficacy of automatic path tracking in virtual colonoscopy", *Computer Assisted Radiology and Surgery (CARS'01)*, Berlin, June 2001.
- M. Benetiere, V. Bottreau, A. Collet-Billon and T. Deschamps, "Scalable Compression of 3D Medical Datasets using a (2D+t) Wavelet Video Coding Scheme." *Sixth International Symposium on Signal Processing and its Applications (ISSPA'01)*, Kuala Lumpur, Malaysia, August 2001.
- R. Truyen, P. Lefere, S. Gryspeerdt and T. Deschamps, "Speed and robustness of (semi-)automatic path tracking", *Virtual Colonoscopy Symposium*, Boston, October 2000.
- T. Deschamps, L. D. Cohen, "Minimal Paths in 3D Images and Application to Virtual Endoscopy", *European Conference on Computer Vision, ECCV'00* (held in Dublin, Ireland, June 2000).
- T. Deschamps, L. D. Cohen, "Path extraction in 3D medical images for virtual endoscopy", *Computer Aided Surgery, Medical Robotics, and Medical Imaging, ISRACAS'00* (held in Haifa, Israel, May 2000).
- T. Deschamps, J.M. Létang, L. D. Cohen and B. Verdonck, "Automatic construction of minimal paths in 3D images for virtual endoscopy", in *Computer Assisted Radiology and Surgery, CARS'99* (held in Paris, France, June 1999).

## Patents:

- T. Deschamps and S. M. Ebeid and L. D. Cohen "Image Processing Method, System and Apparatus for Processing an Image representing a tubular structure and for constructing a path related to said structure", (march 1999).
- M. Greff and O. Gerard and T. Deschamps "Adaptation of potential terms in real-time optimal path extraction", (April 2001).

# Curriculum Vitae

The author was born in Paris, France, on January 19, 1974. After having received general secondary education *(Collège Bartholdi, Boulogne)*, he went to an intermediate school *(Lycée Jacques Prévert, Boulogne)* from which he obtained his "baccalauréat" in 1992. During two years, he endured the preparatory classes for French Engineering schools *(Lycée Claude Bernard, Paris)*, and decided, instead of continuing a third year, to enter university.

In June 1997, he received a Bachelor of Science degree in Mathematics and Computer Science from Paris Dauphine University. And in September 1998, he received a Master of Science with honors in Mathematics, Image Processing and Artificial Intelligence, at CMLA laboratory, Ecole Normale Superieure, Cachan. His graduation project concerned the development of a path tracker for virtual endoscopy in 3D medical image data by means of shortest path techniques and was carried out at the Medical Imaging Systems (MediSys) group of Philips Research France(PRF), in cooperation with EasyVision Advanced Development group of Philips Medical Systems (Best, the Netherlands).

In the subsequent month, he started as a Ph.D. student at the CEREMADE Laboratory, University Paris Dauphine, on a research project concerning the extraction of paths and surfaces in medical imaging using level-sets framework. The project was carried out in the MediSYs Department of PRF (Suresnes, France). The results are described in this thesis.

He was awarded a post-doctoral fellowship, for research on electron microscopy and confocal microscope imaging, to be carried out at the Computing Sciences Division in cooperation with the Life Science Division, Lawrence Berkeley National Laboratory (Berkeley, CA, United States). This project will start in January 2002.

# Extraction de Courbes et Surfaces par Méthodes de Chemins Minimaux et Ensembles de Niveaux. Applications en Imagerie Medicale 3D.

Thomas Deschamps
Medisys - Philips Recherche France, B.P. 301, 92156 Suresnes Cédex, France.

Dans cette thèse nous nous interessons à l'utilisation des méthodes de chemins minimaux et des méthodes de contours actifs par Ensembles de Niveaux, pour l'extraction de courbes et de surfaces dans des images médicales 3D.

Dans un premier temps, nous nous sommes attachés à proposer un eventail varié de techniques d'extraction de chemins minimaux dans des images 2D et 3D, basées sur la résolution de l'équation Eikonal par l'algorithme du Fast Marching. Nous avons montré des résultats de ces techniques appliquées à des problèmes d'imagerie médicale concrets, notamment en construction de trajectoires 3D pour l'endoscopie virtuelle, et en segmentation interactive, avec possibilité d'apprentissage.

Dans un deuxième temps, nous nous sommes interessés à l'extraction de surfaces. Nous avons developpé un algorithme rapide de pré-segmentation, sur la base du formalisme des chemins minimaux. Nous avons étudié en détail la mise en place d'une collaboration entre cette méthode et celle des Ensembles de Niveaux, dont un des avantages communs est de ne pas avoir d'à priori sur la topologie de l'objet à segmenter. Cette méthode collaborative a ensuite été testée sur des problèmes de segmentation et de visualisation de pathologies telles que les anévrismes cérébraux et les polypes du colon.

Dans un troisième temps nous avons fusionné les résultats des deux premieres parties pour obtenir l'extraction de surfaces, et des squelettes d'objets anatomiques tubulaires. Les squelettes des surfaces fournissent des trajectoires que nous utilisons pour déplacer des cameras virtuelles, et nous servent à définir les sections des objets lorsque nous voulons mesurer l'étendue d'une pathologie. La dernière partie regroupe des applications de ces méthodes à l'extraction de structures arborescentes. Nous étudions le cas des arbres vasculaires dans des images médicales 3D de produit de contraste, ainsi que le problème plus difficile de l'extraction de l'arbre bronchique sur des images scanners des poumons.

**Mots clés :** *Chemins minimaux, modèles déformables implicites, segmentation, imagerie médicale 3D, méthodes variationnelles, Level-Sets, Fast-Marching.*

# Curve and Shape Extraction with Minimal Path and Level-Sets techniques. Applications to 3D Medical Imaging.

In this thesis, we focus on the use of minimal path techniques and Level-Sets active contours, for curve and shape extraction in 3D medical images.

In the first part of thesis, we worked upon the reduction of the computing cost for path extraction. We proposed several path extraction algorithms for 2D as well as for 3D images. And we applied those techniques to real medical imaging problems, in particular automatic path extraction for virtual endoscopy and interactive and real-time path extraction with on-the-fly training.

In the second part, we focused on surface extraction. We developed a fast algorithm for pre-segmentation, on the basis of the minimal path formalism of the first part. We designed a collaborative method between this algorithm and a Level-Sets formulation of the problem, which advantage is to be able to handle any topological change of the surfaces segmented. This method was tested on different segmentation problems, such as brain aneurysms and colon polyps, where target is accuracy of the segmentation, and enhanced visualization of the pathologies.

In the last part of the thesis, we mixed results from previous part to design a specific method for tubular shape description and segmentation, where description is the extraction of the underlying skeleton of our objects.

The skeletons are trajectories inside our objects, which are used as well for virtual inspection of pathologies, as for accurate definition of cross-sections of our tubular objects. In the last chapter we show applications of our algorithms to the extraction of branching structures. We study the vascular tree extraction in contrast enhanced medical images, and we apply the same principle to the more complex problem of the bronchial tree extraction in multi-slice CT scanners of the lungs.

**Keywords:** *Minimal Paths, implicit deformable models, segmentation, 3D medical imaging, variational methods, Level-Sets, Fast-Marching.*