UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Dottorato in Scienza e Tecnologia dell'Informazione – XXII Ciclo

Tesi di dottorato / Ph.D. Thesis

Dynamic shape detection and analysis of deformable structures in biomedical imaging

Ph.D. Advisor: Professor Angelo Cenedese

School Coordinator: Professor Matteo Bertocco

Ph.D. Candidate: Alberto Silletti

Dynamic shape detection
and analysis of deformable structures
in biomedical imaging

Copyright 2007 - 2009
by
Alberto Silletti

RINGRAZIAMENTI - ITALIANO

Queste lavoro nasce da molti sforzi, da notti passate a scrivere codice e a pensare al perche' "quell'algoritmo non funziona correttamente". Sono stati anni molto appaganti.

Mi trovo a scrivere "i ringraziamenti", ed e' sorprendente come qualsiasi cosa vi si scriva risulti banale e non rifletta mai quello che si vorrebbe esprimere. Ringrazio il mio Advisor, il Prof. Angelo Cenedese, che in questi tre anni mi ha seguito, aiutato, supportato e sopportato. Angelo mi ha dato notevoli idee tecniche e mi ha aiutato a crescere come giovane ricercatore e persona. Apprezzo la sua indole, e la sua incredibile capacita' di focalizzare le energie. Angelo ha investito su di me, e spero che il lavoro fatto insieme sino ad ora lo abbia, almeno in parte, ripagato.

Ringrazio molto il Prof. Enoch Peserico, mio co-Advisor e persona geniale, il cui acume mi ha sorpreso in svariate occasioni.

Un grazie al mio amico e Prof. (on tenure) Alessandro Abate, che ho avuto modo di conoscere durante il mio soggiorno in California. Alessandro mette passione nel lavoro, e la trasmette ai suoi collaboratori. E' un eccezionale giovane Professore, far molta carriera e io un giorno potro' dire "lo conosco"!

Un grazie va anche al Prof. Ruggero Frezza, mio Advisor per i primi mesi del dottorato, persona superlativa sotto tutti i punti di vista. Anche se abbiamo lavorato poco "gomito a gomito", il suo modo di vedere il mondo e il suo modo di agire sono penetrati profondamente in me.

Grazie a Claire J. Tomlin, Jeff Axelrod, Stephane Vincent. In questi anni sono stato circondato da persone eccellenti, e spero (un po' ne sono convinto) che un po' della loro eccellenza sia entrata in me.

Un grazie particolare va ai miei genitori, mamma Fernanda e papa' Gabriele, che mi hanno sempre aiutato. Questo dottorato lo devo a loro, per l'amore che mi hanno sempre trasmesso e trasmettono tutt'ora, per l' educazione impartitami da bambino, per la tranquillita' economica che hanno saputo assicurarmi. Mi hanno messo nelle migliori condizioni possibili e mi hanno sempre lasciato libero di scegliere la mia strada.

Grazie a mia sorella Erika per le molte vacanze in Olanda, per volermi e avermi sempre voluto bene. Grazie ad Arno, per i molti cappuccini e le discussioni tecniche su Lua, web 2.0, php, videogames e film. Sopratutto, Arno mi ha sempre considerato molto piu' che un cognato.

Un grazie speciale a Luisa, ragazza straordinaria che con il suo amore, le sue mille attenzioni e premure riempie la mia vita. Grazie Luisa per essere venuta, quel giorno, al mare!

Infine un ringraziamento ai miei amici: a Pier Mattia, amico di lunghissima data e compagno di tante avventure, scalate e biciclettate. Quello che sono lo devo anche a lui, a al suo modo di affrontare le cose. Ogni volta rimando stupefatto da come

Pier Mattia arrivi sempre dove vuole arrivare. Un ringraziamento a Marco Bressan, peak-oiler e dottorando come me, per le infinite discussioni notturne sull'esaurimento delle risorse minerarie e per tutti gli scherzi fatto a Massimo Mastromatteo. Grazie a Matteo Negri, amico e coinquilino, per le piccole perle di saggezza.

Grazie ai miei colleghi (di dottorato e non) che hanno reso leggeri questi anni: Maura, Andrea, Francesco, Martina, Ruggero, Stefano, Lucia, Simone, Mirco, Giulia, Mattia, Saverio, Enrico, Gian Antonio, Giulio, Elisa.

THANKS - ENGLISH

This work bears from many efforts, from “coding-nights”, thinking “why doesn’t the algorithm work”?. Indeed, these were wonderful years.

It’s amazing how difficult writing this section is, how difficult finding the right words. Thanks to my Advisor, Prof. Angelo Cenedese. During these three years he followed, helped, supported and go through me. Angelo gave me great technical ideas and helped me in growing as young researcher and person. I like his mood and the ability to focus his energy. Angelo invested on me: I hope my work repaid him, at least partially.

Thanks to Prof. Enoch Peserico, co-Advisor and sharp person. His cleverness surprised me many times.

A special thanks to my friend Prof. on tenure Alessandro Abate, whom I meet during my time in California. Alessandro works with passion and transmit this passion to his collegues. He’s an exceptional young Professor, I’m sure he’ll be a career man and I will be able to say “I know him”!

Thanks also to Prof. Ruggero Frezza, my Advisor during the first months of my PhD, superlative person in all fields. We worked together only for a short amount of time, but his vision of the world and his behavior went deep into me.

Thanks to Claire J. Tomlin, Jeff Axelrod, Stephane Vincent. During these years I was surrounded by excellent persons, and I hope (a bit I believe so) that a little of their excellence came into my possession.

A very special thank goes to my parents, they always helped me. I took this degree because of them, because of the education they gave me when I was a child, because of the love they transmitt and always transmitted, because of the economical stability they provided me. From all the viewpoints, they put me in the best ever conditions and let me free to choose my way.

Thank to my sister Erika, for the many holidays in the Nethederland, for loving me. Thank to Arno, for the many cappuccinos and the many technical discussions on Lua, web 2.0, php, videogames and movies. Above all, Arno has always been keen on me, more than a brother-in-law.

To Luisa, wonderful girl. Her love, her caring and many attentions fill my life. Thank you Luisa, for coming to the seaside that day!

Finally, thanks to all the friends: to Pier Mattia, close and bosom friend, team-mate of many adventures, mountain ascendant and bike ride. I am what I am because of him as well, because of the way he faces things. I have always been dazzled considering how he get everything he wants. A thank to Marco Bressan, peak-oiler and PhD student like me, for the endless nocturne discussions about the depletion of the mineral resources and for the many jokes to a Massimo Mastromatteo. Thanks to Matteo Negri, friend and roomate, for the words of wisdom.

A thanks to my colleagues that makes easy these years: Maura, Andrea, Francesco,

Martina, Ruggero, Stefano, Lucia, Simone, Mirco, Giulia, Mattia, Saverio, Enrico, Gian Antonio, Giulio, Elisa.

FOREWORDS

We are not even close. We are so far from developing efficient and robust Computer Vision and Image Analysis algorithms that we could compare ourselves to children learning basic arithmetic. I use to think to the field as close to the strong AI: after all, 90% of our brain is needed to process the visual signals from the eyes.

During these years as a PhD students, I have implemented a number of applications and algorithms, and in the end I got a sort of "magical power": I can now foretell what is good, what solution will work, and what will probably never do. This work discusses a portion of my whole work, namely the most important works, and, above all, *why* those work well. Asking the right question is the goal to find the right answer, so I asked myself: "Why does this work so well?", or "Why doesn't this work as expected?"

The theoretical framework developed along the chapters answers these questions. The thesis provides a unified approach and provide the reader a tool to attack the problems.

STRUCTURES AND AIMS OF THIS WORK

The goal of this work is to discuss a three-step-approach to *detect*, *analyze* and *synthesize* a shape given an image, or a sequence of images. Chapter 1 discusses what is a *shape*. The concept of shape is fuzzy: before delving with more complex topics we need at least to agree on what we call "shape". Chapter 2 briefly present the biological case studies we used along the work.

Chapter 3 deals with the single shape detection problem, the easiest problem one could face: given a single image with a single shape of interest, how do we design an algorithm to detect it? Chapter 4 extends the single shape detection approach to reticular shapes, a kind of shapes common in biological images. Chapter 5 extends the single shape detection approach (and its reticular analogous) to a sequence of images, exploiting the temporal coherence.

Chapter 6 analyzes the shape, developing new metrics and measures, while chapter 7 closes the work dealing with the synthesis step.

A special section is chapter 8, which covers the Toolbox we developed to detect shapes. The Toolbox is meant to provide functions reusable on a plethora of problems.

Conclusions and future works are discussed in chapter 9.

Contents

1

Shape Detection, Shape Analysis, Simulation

"What is a shape?" Before delving with complex topics, the work introduces the notion of shape and the problem of its definition, investigating a number of interpretations proposed over the years. A three step paradigm (shape detection; shape analysis; synthesis and simulation) is developed to study biological structures. The paradigm recurs along the whole work and some examples are briefly provided. The chapter is organized as follows: section 1.1 investigate the concept of *shape*, section 1.2 introduces the 3 step paradigm and section 1.3 closes the chapter discussing the most popular shape representation models.

1.1 What is a Shape?

The starting point of this work is the concept of *shape*. The term is fuzzy and not uniquely defined as many definitions occurs over scientific and common literature. Common knowledge refers to shape as the part of the space occupied by an object, as determined by its external boundary, abstracting from properties such as colour, content, and material composition, as well as from the object's other spatial properties (position and orientation in space, size).

A shape is the characteristic surface configuration of a thing; an outline or contour; something distinguished from its surroundings by its outline (Farlex Dictionary)

The spatial arrangement of something as distinct from its substance; "geometry is the mathematical science of shape" (WordReference.com)

Sometimes it's easier to define a shape in terms "what it is not":

The form of an object - how it is laid out in space, not what it is made of, or where it is

George Kendall, mathematician and statistician, provided a more comprensive definition taking into account interior:

A shape is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object

Providing a rigorous definition of shape is thus a difficult problem, comparable to what philosopher St. Agostine says about time:

What is time? If you don't ask me, I know what it is. If you ask, I cannot answer

Indeed, during the last decade the definition proposed by George Kendall got mildly accepted, as discussed in [40]

A plane shape $A \in R^2$ has a 1-dimensional side given by features of its boundary $C = \partial A$; and a 2-dimensional side given by its interior. No successful theory of shape description can ignore one or the other

In the following we denote any object of interest and its shape respectively as X and $\Phi(X)$: the process of extracting $\Phi(X)$ from X is referred along this work as shape intuition. Unfortunately even this is a fuzzy concept, rooting its basis in the pattern recognition capability of the humans. It's deeply wired inside the brain and a very dark matter to investigate. Thus, a rigorous mathematical definition of both shape and intuition is beyond the purposes of this work, still we refer to them grounding on the common knowledge and the abstract ideas everyone has.

1.2 Detection, analysis, synthesis, simulation

The study of any biological structure flows along a four step approach: *shape detection*, *shape analysis*, *synthesis* and *simulation*.

Shape intuition, as abstractly defined in section 1.1 takes place from sensors detecting X. In other words, we glance the shape of an object not directly from X, but from the image of X impressed on our retina through the eyes (the sensing devices).

The world as Will and Representation (Arthur Schopenhauer)

The intuition process thus takes the form

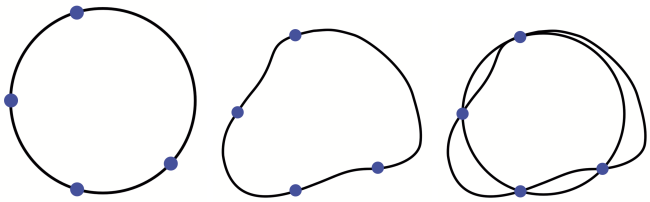
$$X \xrightarrow{sensors} I \xrightarrow{intuition} \Phi(X) \tag{1.1}$$

where I is the sensors' output.

The *representation* step provides an analytical model $\mathcal{R}$ for $\Phi(X)$. The choice is here crucial and greatly influences the next phases. On one side, the representation model grants concreteness and allows for computational algorithms; on the other side it discards information.

$$X \xrightarrow{sensors} I \xrightarrow{intuition} \Phi(X) \xrightarrow{representation} \mathcal{R} \tag{1.2}$$

As an extreme example consider figure 1.1a, representing a circle shape as a set of markers. Figure 1.1b represents the same (deformed) circle. This representation



(a) (b) (c)

Figure 1.1: **Landmarks defined shapes**. *The choice of an analytical model $\mathcal{R}$ for $\Phi(X)$ is a crucial step and largely affects the successive computation. For instance, the figure shows two different landmark defined shapes (a and b). If the landmarks are too coarse, much information is lost and as result it may be impossible to assert the original difference (c)*

model discards information, and thus it's very likely to end up with a distance metric such that $\Phi(X_a) \neq \Phi(X_b)$. This opens a relevant point: while the representation step does not explicitly define a metric space (it does not explicitly define a cross product operator nor a distance metric), it limits the set of spaces and skew the definition of the distance metric towards "specific" functions. In general, an ideal representation should guarantee

$$X_a = X_b \Leftrightarrow \Phi(X_a) = \Phi(X_b) \tag{1.3}$$

For practical cases, this never holds.

A number of shape representation models $\mathcal{R}$ has been proposed over the years, for instance C^2 curves, implicit functions, landmarks based-shapes, constructive geometry ect. As a general advice, note that representing an object as a closed C^2 curve rather than a union of boxes (constructive geometry) or a network graph may lead to very different results. In a nutshell, the representation models are not equivalent. Section 3 deals with this extensively.

Shape detection is the name of the overall process including sensors acquisition, shape intuituion and shape representation.

$$X \xrightarrow{sensors} I \xrightarrow{intuition} \Phi(X) \xrightarrow{representation} \mathcal{R} \tag{1.4}$$

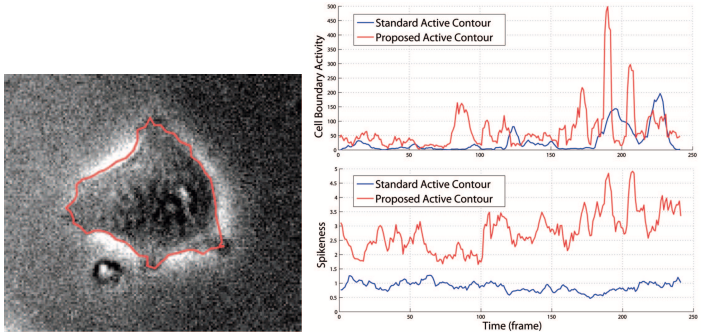
$$X \xrightarrow[\Phi(X)]{detection} \mathcal{R} \tag{1.5}$$

In this work, being the input in form of digital images, *shape detection* mainly involves computer vision and image processing techniques. Indeed, our formulation is general and does not assume any peculiar sensor input nor any representation format.

Shape analysis takes as input a shape representation model and analyses it, resulting in measures.

$$X \xrightarrow[\Phi(X)]{detection} \mathcal{R} \xrightarrow{analysis} R^k \tag{1.6}$$

R^k is a "set of number", coding the output of the analysis. These numbers could be forces, lengths, labels, descriptors, medical diagnosis, classifications etc. The type of analysis is task dependent and so is the high level interpretation of these numbers.



(a) (b)

Figure 1.2: **Miocardial cell activity and spikeness**. *A miocardial cell (a) is outlined in a video sequence using a C^2 contour. (b) shows the graphs of the cell activity and spikeness over time as directly computed from the C^2 contour*

For instance we analyzed the deformation of the cells using the *deformotion* approach [60] and extract relevant metrics such as central moments spikeness and cell activity (fig 1.2). This is crucial to assess the cell vitality and to distinguish between different type of cells. In melanocytic lesions (fig. 1.3), we first compute metrics such as area, color and perimeter, and then proceed to a comparison over different segmentation algorithms. In the drosophila epithelium, we compute the stress and the deformation of the epithelium cells over time, correlates the vertices, and infer dynamics.

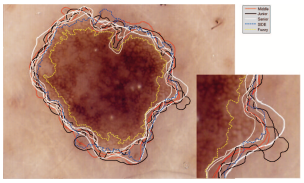


Figure 1.3: **A melanocytic lesion**. *Melanocytic lesions, or "Banal nevus", or "Nevocytic nevus", have to be constantly monitored by dermatologists. During the lifetime, an adult can grow over* 500 *of these lesions*

Synthesis provides a dynamical model for X. This involves the understanding of the high level behavior of the object. The goal is to simplify the observed X and to discard noisy and irrelevant information. While the representation model $\mathcal{R}$ is close to the data (it is basically an analythical form of $\Phi(X)$), the synthesis model $\mathcal{M}$ is close

to the real object X. It may for instance be a mechanical model of X (with masses, springs and dampers), a probabilistic model, a Markov chain, etc.

$$X \xrightarrow[\Phi(X)]{detection} \mathcal{R} \xrightarrow{synthesis} \text{Model}\mathcal{M} \tag{1.7}$$

This work presents a probabilistic modelization of melanomas and the creation of a "prototype". We also introduce a mechanical model for drosophila morphogenesis, inspired to cloth simulation.

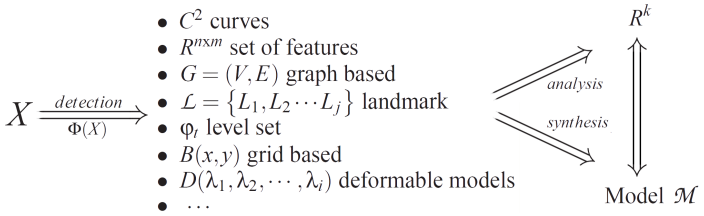


Figure 1.4: **The complete view**. *Detection, Shape Representation Models, Analysis and Synthesus as discussed along this work*

1.3 Shape representation

A shape representation, as discussed in section 1.2, is a suitable model to represent the abstract concept of shape $\Phi(X)$. It is a model in the sense that it "models" the category of shape to belong to some predefined configuration, such as continuous surfaces or reticular grids. Thus, the shape representation is the first necessary abstraction layer, yet pretty close to the data but still a modelization.

Many shape representation models have been proposed over the years. The majority of them are born from practical problems rather than from theoretical speculations, thus in many cases each representation has its own strengths (usually crucial for the practical application) and drawbacks (usually negligible).

- C^d **hypersurfaces**. The shape is embedded in a d-dimensional space as a $C^d(s_1, s_2, \cdots, s_d)$ continuous surface, where $C^d \in R^d$ and $s_i \in [0,1]$ for $i = 1, \cdots, d$ [34]. C^d is constrained to be continuos such that $\lim_{s_i \to \hat{s}_i} C^d(s_i) = C^(\hat{s}_i)$. Smoothness is not enforced; Shapes can be closed or open as well. C^2 curves and C^3 surfaces (figure 1.5) are very common examples [59, 48], where respectively $C^2 \in R^2$, $s_1 = s \in [0,1]$ and $C^3 \in R^3$, $s_1 = s \in [0,1]$, $s_2 = t \in [0,1]$. Spline curves and spline surfaces, popular in computer graphics, are common implementation. The form for C^d is here analythically provided by the Bezier curve theory which defines a point C for every $s_i \in [0,1]$.

- R^{nxm} **shape functions**, sometimes also called "contour functions", model $\Phi(X)$ as a set of (n) m-dimensional functions [35, 45]. Each function is pointwise

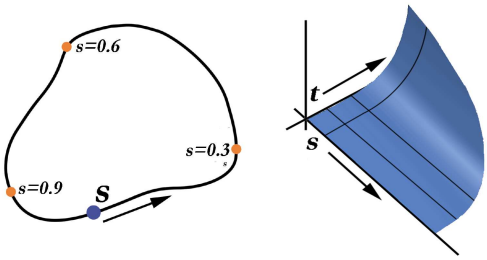


Figure 1.5: C^2 **and** C^3 **surfaces**. *C^2 and C^3 surfaces are very popular in computer graphics to model paths and solids. Bezier and Spline curves [7, 8] provide an easy framework to implement them and obtain smooth surfaces. The natural parametrization of the curve in (s,t) allows also for easy texturing*

related to the shape and describe a peculiar aspect of the contour in that point. Popular shape functions are for instance distance from the centroid ("radius-vector function"), angle-tangent function, integral of the contour length, etc. Figure 1.6 shows an example of radius-vector.

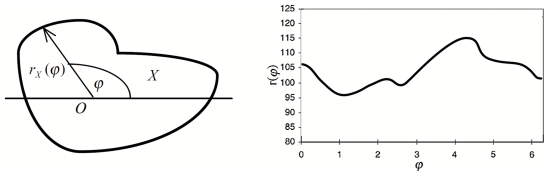
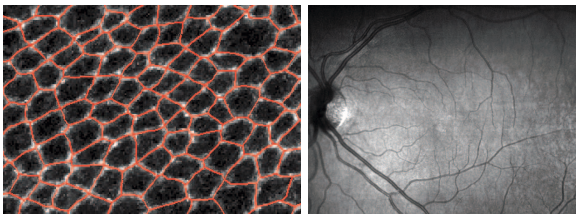


Figure 1.6: **Radius Vector function**. *The Radius Vector $r_x(\varphi)$ originates from the centroid of the shape and intersects $\Phi(X)$ within a certain angle φ. The length (magnitude) of the vector defines the scalar value of the Radius Vector function in terms of* φ

- A $G=(V,E)$ **graphs** model a shape using relevant points (vertices V) and their relationships (edges E). Weighted edges usually represents geometric relationships (such as euclidean point distance) while vertices may be coupled with absolute coordinates to locate them in a reference space [15, 16, 17]. Graph based shapes straightforward model reticular structures such as cellular ephitelium or blood vessels on retina (fig 1.7)

- In the **landmark** approach, one assumes that the shape of an object can be represented by the coordinates of a set $\mathcal{L}=\{L_1,L_2\cdots L_j\}$ of landmarks, which are points of particular interest. "Anatomical" landmarks designate part of the organism corresponding to biologically meaningful location (e.g. the corner of an eye, the tip of the fingers). "Mathematical" landmarks are points corresponding to some mathematical or geometrical property (points of high curvature, extreme points, etc.). The landmark shape representation is popular in motion



(a) (b)

Figure 1.7: **Reticular structures**. *Nature presents a wide spectrum of reticular structures, aring mostly from cells packing or from vascular systems. (a) shows the drosophila's wing epithelium, with a white protein marker highlighting the borders. A graph, in red, captures the cellular structure. (b) shows the blood vessels on a retina image*

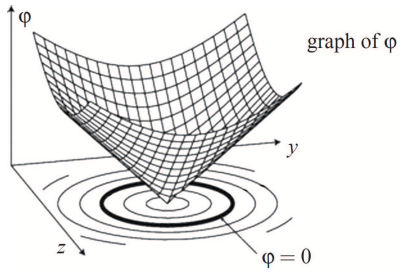


Figure 1.8: **Implicit function** $\varphi(y,z) = y^2 + z^2 - 1$. *An implicit function* $\varphi(y,z)$ *is a scalar function that defines a shape as the set of all points* $(y,z) \in R^2$ *such that* $\varphi(y,z) = 0$. *Implicit functions are widely used in the Level Sets approach [48], in which* $\varphi(y,z)$ *changes in time to reflect the changes of the shape*

capture techniques, where markers are physically attached to a body and tracked via cameras to extract the shape dynamics.

- φ **implicit functions** model a shape as the zero-isocontour of a function φ [42]. Suppose you define a function $\varphi(x)$, x beloning to some domain (for instance $x \in R^2$) and φ a scalar real function. The zero isocontour is the set of all points x where $\varphi(x) = 0$. To fix ideas, consider $\varphi(y,z) = y^2 + z^2 - 1$. The shape is defined by the $\varphi(y,z) = 0$ isocontour, which is the boundary of the unit sphere defined as $\{x : |x| = 1\}$. More generally, in R^n the implicit function $\varphi(x)$ is defined over all $x \in R^n$, and its isocontour has dimension $n-1$. Initially the implicit representation might seem wasteful, since the implicit function is defined on all of R^n while the contour has only dimension $n-1$. However, a number of powerful tools are here available. Adding dynamics to implicit functions is for instance natural and leads to the level sets and the fast marching method, widely used in computer graphics to simulate fluids.

- The **volume element representation (voxel)**. $V(x)$ models the space embedding the object X as a discrete space (a grid of square elements, voxels). Each voxel represents a value in the space, according to some criterions. For instance, one can say that the voxels totally inside the object are "inside", while all the other are "outside". As with pixels, voxels themselves typically do not

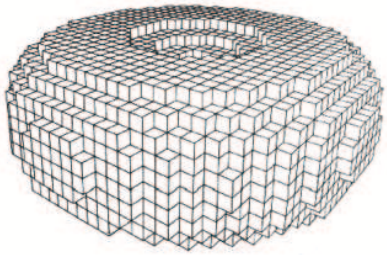
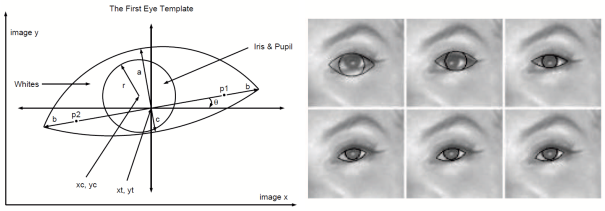


Figure 1.9: **Torus Voxels representation**. *A voxels model can represent almost any shape with a tunable degree of quality. Voxels are also widely used to represent non-solid, "fluid", objects such as smoke, electron cloud and air*

  contain their position in space, but rather it is inferred on their position relative to other voxels. Voxel representation is suitable to model blurry objects, such as electron clouds, or smokes, where each voxel is marked with the probability of the object (or with the density of the object) opposed to a simple inside/outside representation.

- A **deformable model** (or template) $D(\lambda_1, \lambda_2, \cdots, \lambda_k)$ is a standard shape for a class of objects [7]. The template contains a number of parameters λ_i and a cost functional $c(\lambda_1, \cdots, \lambda_k)$ specifing how good the customization of the template fits the data. The goal is to tune the parameters to minimize the functional. Deformable models, explicitely enforcing the membership class of an object, have been extensively applied in the "expert vision" systems, where strong knowledge is available and the class of the shapes to be recognized is precisely defined (fig. 1.10)

- **Constructive Solid Geometry (CSG)** allows to define a complex shape by using boolean operators on other CSG shapes or primitives [20]. A primitive is the simplest possible solid object, typically prisms, cylinders, pyramids, spheres and cones. Combining an (infinite) number of primitives virtually leads to any possible shape, whereas this is practically limited by computation efficiency. Constructive geometry has many practical uses. It is used in cases where simple geometric object are desidered, or where mathematical accuracy is important. Natively CSG assure that objects are "solid" or water-tight if all primitive are water-tight. This can be important for some manufacturing or engineering applications. By comparison, when creating a shape based upon boundary representations, additionals checks must be performed to assure continuity and



(a) (b)

Figure 1.10: **Deformable model of an eye**. *Deformable models greatly restricts the space of possible shapes imposing a "model" of appearance, generally defined by few paramentes. (a) for examples shows an eye template in terms of a modest number of geometric parameters. (b) In successive iterations of a gradient descent algorithm, an equilibrium configuration is reached in which the template fits the eye closely. (reprinted from [61])*

to prevent "holes". Furhermore, a point in space can be easily tested against the shape to determine whatever it is inside, outside or on the boundary. This is a desiderable quality for some applications such as collision detection (for instance in computational physics).

2

Case Studies

The chapter provides a brief understanding of the most important case studies we faced along the work, providing a basic medical / chemical knowledge. In particular, the Drosophila Melanogaster fruit fly, myocardial cells and melanocytic lesion case studies are here presented.

2.1 The Drosophila Melanogaster

Drosophila melanogaster (Greek for dark-bellied dew lover) is a species of *Diptera*, or the order of *flies*, in the family *Drosophilidae*. The species is commonly known as the common fruit fly or vinegar fly. This species is one of the most commonly used model organisms in biology, including studies in genetics, physiology, microbial pathogenesis and life history evolution because they are easy to take care of, breed fast, and lay many eggs.

Drosophila is so popular, it would be almost impossible to list the number of things that are being done with it. Originally, it was mostly used in genetics, for instance to discover that genes were related to proteins and to study the rules of genetic inheritance. More recently, it is used mostly in developmental biology, looking to see how a complex organism arises from a relatively simple fertilised egg. Embryonic development is where most of the attention is concentrated, but there is also a great deal of interest in how various adult structures develop in the pupa, mostly focused on the development of the compound eye, but also on the wings, legs and other organs. Among the main reasons that determined the "success" of the drosophila:

- the care and culture requires little equipment and use little space even when using large cultures, and the overall cost is low
- it is small and easy to grow in the laboratory and their morphology is easy to identify once they are anesthetized

- it has a short generation time (about 10 days at room temperature) so several generations can be studied within a few weeks
- it has a high fecundity (females lay up to 100 eggs per day, and perhaps 2000 in a lifetime)
- its complete genome was sequenced and first published in 2000. [2]

Figure 2.1: **Male and female adult Drosophila Melanogaster**. *Males (left) are smaller than females (right). In Drosophila melanogaster, not only the females are larger than males for most body dimensions but also the sexes differ in pigmentation, the number of visible abdominal segments, structure of the genitalia, presence of sex combs, shape of various body parts, behavior and numerous other features*

Figure 2.2: **Adult female specimen**. *A* 2.5 *x* 0.8 *mm Drosophila melanogaster fly*

Wildtype fruit flies have brick red eyes, are yellow-brown in color, and have transverse black rings across their abdomen. Males are easily distinguished from females based on color differences, with a distinct black patch at the abdomen, less noticeable in recently emerged flies, and the sexcombs (a row of dark bristles on the tarsus of the first leg). Furthermore, males have a cluster of spiky hairs (claspers) surrounding the reproducing parts used to attach to the female during mating.

The developmental period for Drosophila melanogaster varies with temperature, as with many ectothermic species. The shortest development time (egg to adult), 7 days, is achieved at 28 C. Females lay some 400 eggs (embryos), about five at a time. The eggs, about 0.5 millimetres long, hatch after 1215 hours. The resulting larvae grow for about 4 days while molting twice (into 2nd- and 3rd-instar larvae), at about 24 and 48h after hatching. During this time, they feed on the microorganisms that decompose the fruit, as well as on the sugar of the fruit itself. Then the larvae

encapsulate in the puparium and undergo a four-day-long metamorphosis, after which the adults eclose (emerge).

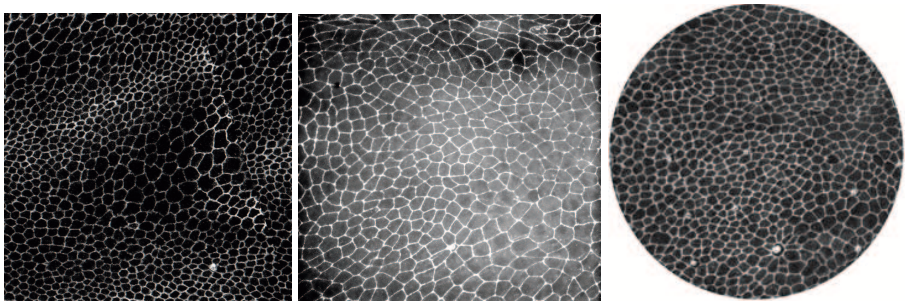


Figure 2.3: **A patch of Drosophila epithelium**. *Drosophila epithelium as viewed from a confocal laser scanning microscopy over time, during the embryo developmental stage. In white: one can track E-cadherin protein tagged with a White Fluorescent Protein, which plays an important role in cellular adhesion*

Fluorescence optical microscopy has become an essential imaging technique for research in biology, especially in the field of developmental biology. Recent technological progress have enabled the development of faster microscopes offering higher resolution to collect images of living biological samples. In parallel, important progress have been made in the development of stable and bright fluorescent probes. A confocal microscopy gave us a fourty-frames-long video sequence form the embryo development stage (see image 2.3). Cells are highlighted in white by means of a protein marker.

2.2 Myocardial cells

Myocardial cells pertains to the muscular tissue of the heart (the myocardium). Diseases to myocardium infarction are a common cause of death in developed countries. The initiation and evolution of cardiac failure depends on the accumulation of old, poorly contracting cells. When myocardium is damaged by injury, such as a heart attack, the functional contracting heart muscle dies and is replaced with nonfunctional scar tissue.

Heart transplantation is currently the last resort for end-stage heart failure, but is hampered by a severe shortage of donor organs and rejection. Tissue engineering and cell-based therapies have been recently proposed as promising alternative. In this context, cellular transplantation and tissue engineering approaches have emerged as promising alternatives to heart transplantation. Damaged cells could be replaced with healthy ones, providing an endless lifetime for the whole tissue. Either, the whole tissue could be cultivated in vitro and then transplanted.

In details, the former approach involves the transplantation of isolated cells directly in the injury. Various types of cells have been considered for the repair of

damaged myocardial tissue such as fetal cardiomyocytes, embryonic stem cells, bone marrow derived stromal and mesenchymal stem cells ([36] [21]). However, cell transplantation has several disadvantages, including substantial cell death soon after the injection of cells. In addition, cell orientation and electromechanical connections after cell engraftment are only partially controllable. The latter approach to cardiac regeneration could be the myocardial implantation of tissue created in vitro ([46] [43]). The aim of an engineered cardiac graft is to provide a large source of viable donor cells to repopulate the myocardium or to provide directly a contracting tissue to replace an injured myocardial area. However, the high mechanical stress represented by the permanent cardiac contraction/relaxation cycle, as well as the electric integration of the tissue within the native cardiac muscle, add to the complexity of myocardial tissue engineering.

Figure 2.4 shows typical cultures as view from a microscope.

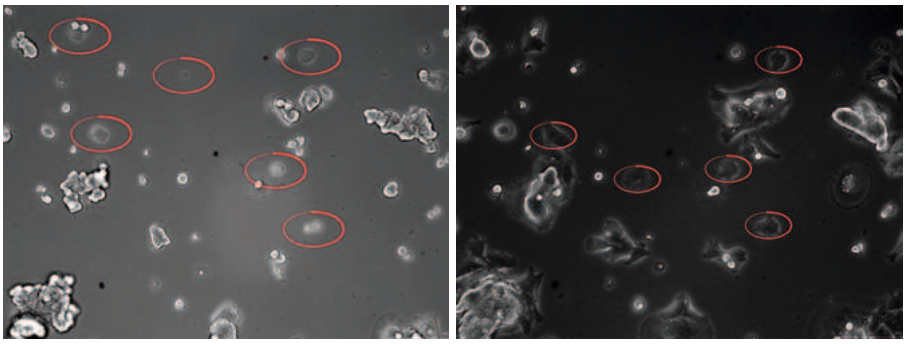


Figure 2.4: **A vitro of myocardial cells**. *Myocardial cells are circled in red. The white spots are mainly air bubbles. Cells adhere to the glass and becomes sort of transparente*

Selecting the type of cells and creating a suitable environment in which cells can grow and organize themselves in a functional way are thus foundamental problems, technically and biologically. Numerous studies have experimentally addressed the potential of different types of stem cells to differentiate in contractile cells. Stem cells are self-renewing and undifferentiated primitive cells that develop into functional, differentiated cells. After differentiation, these cells should integrate both functionally and structurally into the surrounding viable myocardium and develop a network of capillaries and larger size blood vessels for supply of oxygen and nutrients to the injured region. In this direction, “Biomaterials” are used to provide structural support during the initial stages of tissue formation. The cells utilize the biomaterial as a support for initial attachment and remodeling, but then slowly begin to generate their own components and grow independently.

The *in vitro* characterization of a single cell physiology is of primary importance both to evaluate the mature differentiation of stem cell toward cardiac lineage both to select and grow only to most promising cells. The complete differentiation to cardiomyocyte is characterized not only by the expression of mature “cardiac markers”

(proteins such as MHC, actinin, desmin, ANP or troponin) but also by the exhibition of spontaneous and rhythmic contractions. In this context, the possibility of extrapolating automatically relevant physiological information about the cell or tissue contraction from optical analysis, offers an interesting and sensitive tool to measure online the cardiac functionality.

2.3 Melanocytic lesion

A melanocytic lesion, also known as a “nevus”, is a type of lesion that contains nevus cells. Some sources equate the term “mole” with “melanocytic nevus” [44]. Other sources reserve the term “mole” for other purposes.

According to the American Academy of Dermatology, the majority of moles appear during the first two decades of a persons life, while about one in every 100 babies is born with moles. Acquired moles are a form of benign neoplasm, while congenital moles, or congenital nevi, are considered a minor malformation or hamartoma and may be at a higher risk for melanoma. A mole can be either subdermal (under the skin) or a pigmented growth on the skin, formed mostly of a type of cell known as a melanocyte. The high concentration of the bodys pigmenting agent, melanin, is responsible for their dark color.

The most common variants of nevus are:

- *Dysplastic nevus* (nevus of Clark): usually a compound nevus with cellular and architectural dysplasia. Dysplastic nevi can be flat or raised. While they vary in size, dysplastic nevi are typically larger than normal and tend to have irregular borders and irregular coloration. Hence, they resemble melanoma, appear worrisome, and are often removed to clarify the diagnosis. Dysplastic nevi are markers of risk when they are numerous (atypical mole syndrome). According to the National Cancer Institute (NIH), doctors believe that dysplastic nevi are more likely than ordinary moles to develop into the most virulent type of skin cancer called melanoma.

- *Blue nevus* is blue in color as its melanocytes are very deep in the skin. The nevus cells are spindle shaped and scattered in deep layers of the dermis. The covering epidermis is normal.

- *Spitz nevus* is a distinct variant of intradermal nevus, usually in a child. They are raised and reddish (non-pigmented). A pigmented variant, called the nevus of Reed, typically appears on the leg of young women.

- *Acquired nevus* is generically any melanocytic nevus that is not a congenital nevus or not present at birth or near birth.

It often requires a dermatologist to fully evaluate moles. For instance, a small blue or bluish black spot, often called a blue nevus, is usually benign but often mistaken

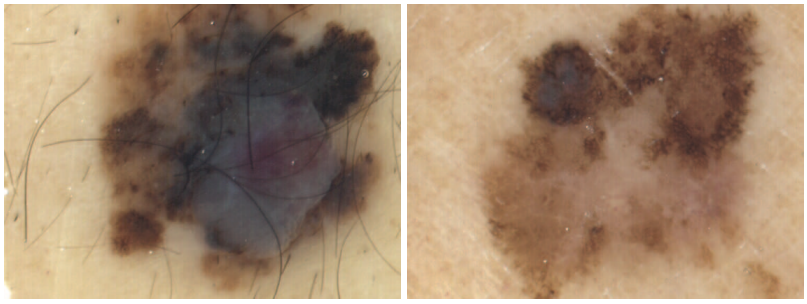


Figure 2.5: **Blue and Dysplastic naevi**. *A Blue nevus (left) shows the typical blue halos. A Dysplastic nevus (right) displays irregular border and coloration*

for melanoma. Conversely, a junctional nevus, which develops at the junction of the dermis and epidermis, is potentially cancerous.

A basic reference chart used for consumers to spot suspicious moles is found in the mnemonic A-B-C-D, used by institutions of dermatology The letters stand for Asymmetry, Border, Color, and Diameter. Sometimes, the letter E (for Elevation or Evolving) is added. If a mole starts changing in size, color, shape or, especially, if the border of a mole develops ragged edges or becomes larger than a pencil eraser, it would be an appropriate time to consult with a physician. Other warning signs include a mole, even if smaller than a pencil eraser, that is different than the others and begins to crust over, bleed, itch, or becomes inflamed. The changes may indicate developing melanomas. The matter can become clinically complicated because mole removal depends on which types of cancer, if any, come into suspicion.

A recent and novel method of melanoma detection is the “Ugly Duckling Sign” [38] It is simple, easy to teach, and highly effective in detecting melanoma. Simply, correlation of common characteristics of a person’s skin lesion is made. Lesions which greatly deviate from the common characteristics are labeled as an Ugly Duckling, and further professional exam is required. The “Little Red Riding Hood” sign, [38] suggests that individuals with fair skin and light colored hair might have difficult-to-diagnose melanomas. Extra care and caution should be rendered when examining such individuals as they might have multiple melanomas and severely dysplastic nevi. A dermatoscope must be used to detect “ugly ducklings”, as many melanomas in these individuals resemble non-melanomas or are considered to be “wolves in sheep clothing”. These fair skinned individuals often have lightly pigmented or amelanotic melanomas which will not present easy-to-observe color changes and variation in colors. The borders of these amelanotic melanomas are often indistinct, making visual identification without a dermatoscope very difficult.

People with a personal or family history of skin cancer or of dysplastic nevus syndrome (multiple atypical moles) should see a dermatologist at least once a year to be sure they are not developing melanoma.

3

Static Single Shape detection

This chapter addresses the single shape detection problem, which will be used around the work as a basic building block. Section 3.1 presents the problem and gives the theoretical background. Section 3.2 discuss the Active Contours framework, an effective metodology to detect shapes. We extend the framework in section 3.3, building the Generalized Active Contours, which proved to be a more reliable and robust tool. Section 3.4 applies the Active Contours and Generalized Active Contours to a number of case studies, while section 3.5 closes the chapter showing a series of results and experiments.

3.1 Single shape detection

The single shape detection problem is the problem of finding a (single) shape Φ in a digital image I. This is the building block of almost any computer vision system, being its resolution a necessary development step.

Roughly speaking, the problem brings back to a minimization over the set S of all the possibile representation $\mathcal{R}(\Phi_c)$ [7, 52]:

$$\mathcal{R}(\Phi) = \underset{\mathcal{R}(\Phi_c)\in S}{\arg\min}\, \mathcal{E}(I, \Phi_c) \tag{3.1}$$

choosing as optimal $\mathcal{R}(\Phi)$ the one that minimizes the energy functional $\mathcal{E}(I, \Phi_c)$. $\mathcal{E}$ is a scalar function of the (image) data I as well of the candidate shape Φ_c. In the following, to keep notation simple, we'll write Φ for $\mathcal{R}(\Phi)$ where the context is clear. Thus, equation 3.1 becomes

$$\Phi = \underset{\Phi_c\in S}{\arg\min}\, \mathcal{E}(I, \Phi_c) \tag{3.2}$$

Even though the problem is of easy statement, there are two main issues playing a central role.

Firstly, the computation of the global minima is often unfeasible. In other words, but for trivial problems with a tightly defined structure, it's impossible to find a closed form for the minimization or to explore exhaustively S. Depending on the nature of the problem, the cardinality of S could be even infinite [18]. Thus, most often, algorithms find suboptimal solutions. Secondly, the definition of $\mathcal{E}$ is not unique and there are no general guidelines on how to build it. A good $\mathcal{E}$ should be a concave function with a unique minima. Furthermore, $\mathcal{E}$ has to be related to the high-level perception of Φ. In a nuthshell, the minima of $\mathcal{E}$ should be visually recognizable as "good" matches for the shape.

Having an ideal function to search over S does not guarantee the success of the global detection procedure, since also the definition of $\mathcal{E}$ defines the (upper level) quality boundary. Analogously, given a bad exploration procedure of S, a good concave functional $\mathcal{E}$ cannot guarantee good results. These observations suggest that a reliable shape detection algorithm should ground on a good implementation of both the aspects.

3.2 Active Contours

Active Contours, also called Snakes [34], are a framework for delineating an object outline from possibly noisy data. Formally, an Active Contour is a curve

$$\mathcal{C}(s) \in \mathbb{R}^d, \quad s \in [0,1]^d \tag{3.3}$$

evolving in psuedo-time according to an associated energy $\mathcal{E}(\mathcal{C})$:

$$\mathcal{E}(\mathcal{C}) = \mathcal{S}(\mathcal{C}) + \mathcal{P}(\mathcal{C}) \tag{3.4}$$

where

$$\mathcal{S}(\mathcal{C}) = \int_{\partial C} \alpha(s) \left| \frac{\partial \mathcal{C}}{\partial s} \right|^2 + \beta(s) \left| \frac{\partial^2 \mathcal{C}}{\partial s^2} \right|^2 ds \qquad \mathcal{P}(\mathcal{C}) = \int_{\partial C} \mathcal{D}\left[I(\mathcal{C}(s))\right] ds \tag{3.5}$$

The internal energy $\mathcal{S}(\mathcal{C})$ enforces the smoothness of the curve $\mathcal{C}$. This is a high level absumption stating that smooth surfaces in the real world maps to smooth "data" in the sensor. The term $\left|\frac{\partial \mathcal{C}}{\partial s}\right|^2$ drives the final rest shape towards short curves, while the term $\left|\frac{\partial^2 \mathcal{C}}{\partial s^2}\right|^2$ imposes low curvature values. In other words, the former grants high energy to elongated contours (elastic force) and the latter to bended/high curvature contours (rigid force). $\alpha(s)$ and $\beta(s)$ are scalar terms to balance the magnitude of the energy term in each segment of $\mathcal{C}$

The external energy $\mathcal{P}(\mathcal{C})$ is minimal when $\mathcal{C}$ is at the object boundary position. The most straightforward approach consists in choosing $\mathcal{D}[I] = -|\nabla I|$. This drives the final rest shape towards the edges of the image. Again, this is a high level assumption on the nature of the data, enforcing that the object's boundary in the real world maps to points of high derivative in the image.

Active Contours naturally behave according to the framework in 3.1, where $\mathcal{E} = \mathcal{E}(\mathcal{C}) = \mathcal{S}(\mathcal{C}) + \mathcal{P}(\mathcal{C})$ (3.4) and $\mathcal{R}(\Phi) = \mathcal{C}(s)$ (3.3). In this case, the energy function is well-defined and able to cope with a plethora of application simply tuning the parameters $\alpha(s)$ and $\beta(s)$. Furthermore, a large literature has been developed to explore efficiently S, such as inflating and balloning Active Contours [32], Magnetostatic Active Contours [57], geodesic A.C. [11] and GVF A.C. [58, 24]. The combination of these two aspects makes Snakes one of the most effectively used tool in shape detection.

According to the variational principle, forces acting on $\mathcal{C}(s)$ can be derived as:

$$F(s) = \frac{\partial \mathcal{E}}{\partial s} \tag{3.6}$$

$$F(s) = \frac{\partial \mathcal{S}(\mathcal{C})}{\partial s} + \frac{\partial \mathcal{P}(\mathcal{C})}{\partial s} \tag{3.7}$$

thus

$$F(s) = -\nabla \mathcal{D}[I] + 2\frac{\partial}{\partial s}\left(\alpha(s)\frac{\partial \mathcal{C}}{\partial s}\right) + 2\frac{\partial^2}{\partial s^2}\left(\beta(s)\frac{\partial^2 \mathcal{C}}{\partial s^2}\right) \tag{3.8}$$

and the minimum rest shape is reached for:

$$F(s) = 0 \tag{3.9}$$

3.3 Generalized Active Contours

"Standard" Active Contours suffer from a number of major drawbacks.

Firstly, they are not able to cope with images with soft edges or heavy textures. The external energy term $\mathcal{P}(\mathcal{C})$ requires, in fact, edges to be strong and clear in order to attract the contour. This requirement is often unheeded. Furthermore, real world images often show a myriad of edges of similar intensity, causing the snake to pick one of them almost randomly.

Secondly, the exploration of S starts from an initial guess contour and proceeds in a sort of "steepest descendant" way to reach a good minima for $\mathcal{E}(\mathcal{C})$. Thus, the initial configuration is crucial. In absence of an automatic procedure, an operator has to manually initialize the algorithm. The initialization problem lies in the design of the functional $\mathcal{E}(\mathcal{C})$ that is not a concave function (indeed, it is *far* from being a concave function) and thus cannot converge to a global minima.

Lastly, the internal energy functional $\mathcal{S}(\mathcal{C})$ imposes smoothness on the curve $\mathcal{C}$. This is not always the case, since spiky, bouncy, and sharp-corner shapes are very common in nature as well in industrial design.

The proposed Generalized Active Contours are a novel approach. They extends the classical formulation introducing *generalized energy* terms $\mathcal{S}_g(\mathcal{C})$ and $\mathcal{P}_g(\mathcal{C})$, called *model energy* and *image energy*. Formally

$$\mathcal{E}_g(\mathcal{C}) = \mathcal{S}_g(\mathcal{C}) + \mathcal{P}_g(\mathcal{C}) \tag{3.10}$$

and

$$\mathcal{S}_g(\mathcal{C}) = \mathcal{S}_g\left(\frac{\partial \mathcal{C}}{\partial s}, \frac{\partial^2 \mathcal{C}}{\partial s^2}, \mathcal{F}_i\right) \qquad \mathcal{P}_g(\mathcal{C}) = \mathcal{P}_g(\mathcal{D}[I], \mathcal{G}_j) \tag{3.11}$$

where $i = \{1 \cdots n_f\}$, $j = \{1 \cdots n_g\}$ and $\mathcal{F}_i$ and $\mathcal{G}_j$ are additional shape and image energies. In other words, formula 3.11 means that the energy functionals $\mathcal{S}_g(\mathcal{C})$ and $\mathcal{P}_g(\mathcal{C})$ are functions not just of the derivatives of the curve $\mathcal{C}$ and of the image I, but indeed of some supplementary energies $\mathcal{F}_i$ and $\mathcal{G}_j$.

To keep things simple, formula 3.11 can be revised as a straight linear combination:

$$\mathcal{S}_g(\mathcal{C}) = \int_{\partial C} \alpha(s) \left|\frac{\partial \mathcal{C}}{\partial s}\right|^2 + \beta(s) \left|\frac{\partial^2 \mathcal{C}}{\partial s^2}\right|^2 ds + \sum_{i=1}^{n_f} \gamma_i \mathcal{F}_i \qquad \mathcal{P}_g(\mathcal{C}) = \int_{\partial C} \mathcal{D}[I]\, ds + \sum_{j=1}^{n_g} \lambda_j \mathcal{G}_j \tag{3.12}$$

being γ_i and λ_j scalar values playing the same weighting role of $\alpha(s)$ and $\beta(s)$. Note that that formula 3.12 is an over-simplification since it excludes all the (possibly) non linear combination of $\mathcal{F}_i$ and $\mathcal{G}_j$ and, even more, uses the derivatives of $\mathcal{C}$ to impose smoothness constraints.

The term $\mathcal{F}_i$ and $\mathcal{G}_j$ plays a central role. Each of these models a peculiar aspect of the object X and drives the Active Contour $\mathcal{C}$ towards *that* peculiar aspect. Generalized Active Contour provides a unified framework to build these terms in a rigorous and robust way. Furthermore, close in spirit to the aphorism of chapter 1

A plane shape $A \in R^2$ has a 1-dimensional side given by features of its boundary $C = \partial A$; and a 2-dimensional side given by its interior. No successful theory of shape description can ignore one or the other

the terms $\mathcal{F}_i$ and $\mathcal{G}_j$ allow for an easy treatment of the boundaries of $\mathcal{C}$ as well for the interior.

Before delving with the creation of $\mathcal{F}_i$ and $\mathcal{G}_j$, it's importat to remark a conceptual difference between them. The former, $\mathcal{F}_i$, addresses and drives the Active Contour towards distinctive features of $\mathcal{C}$, such as area, length of the perimeter, smoothness of the corners, sphericity of $\mathcal{C}$, central moments, centroid. The latter, $\mathcal{G}_j$, on the opposite drives the Contour towards distinctive features of $I(\mathcal{C})$, such as the internal color, the blur of the edges, the texture, the external neighbor color. Recalling the original distintion between $\mathcal{S}(\mathcal{C})$ and $\mathcal{P}(\mathcal{C})$ introduced in equation 3.4, the terms $\mathcal{F}_i$ are shape-related, while $\mathcal{G}_j$ are data-related.

Suppose now that the object X is characterized by a certain metric τ, providing a single scalar value or a vector. In other words, τ quantitatively "measures" a certain feature of the object. For this reason, we call it a *metric* for X. The first step is to identify such a metric.

Being Γ_τ the probability density function of the metric τ, one could build a related energy term as follow:

$$\mathcal{E}_\tau = -\Gamma_\tau(\tau_c) \quad \Gamma_\tau \in R^+ \quad \Gamma_\tau \leq 1 \tag{3.13}$$

where $\Phi_c \in S$ is a candidate shape, and τ_c is the value of the metric τ on Φ_c. $\Gamma_\tau(\tau_c)$ expresses the likelihood of Φ_c. Low values indicate a low probability for Φ_c to be a good candidate (and thus high energy), whereas high value designate a good fitness (and thus low energy) (see figure 3.1). The probability density function Γ_τ may be derived in closed form analytically, approximated via an object training set, or eventually approximated empirically. In general, for non trivial metric, the closed form is unknown and a learning phase is necessary.

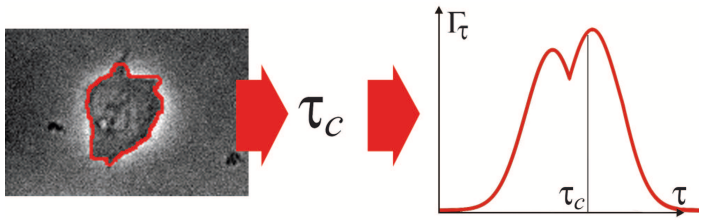


Figure 3.1: **Probability density function for a scalar metric** τ. *The definition of the probability density function* Γ_τ *is the the first step to define a generalize energy term* $\mathcal{E}_\tau$ *In the figure, a good value of* τ_c *delivers a low energy state for the candidate shape* Φ_c*, thus indicating* Φ_c *as a "good" candidate for minimization*

According to its nature, the generalized energy term of equation 3.13 may be one of the terms $\mathcal{F}_i$ or $\mathcal{G}_j$. Practically speaking, there is no big deal in classifying it as one or either term, considering also that some metrics τ may originated energies $\mathcal{E}_\tau$ of non clear classification.

The strength of 3.13 is twofold. Firstly, any relevant metric can be turned in an energy term. The approach does not limit to straighforward metric such as $\left|\frac{\partial C}{\partial s}\right|^2$, $\left|\frac{\partial^2 C}{\partial s^2}\right|^2$ or ∇I; indeed it opens to a possibly very high number of them. This is especially important for noisy and fuzzy images, where little or no information is carried by ∇I.

Secondly, 3.13 provide an easy way to incorporare prior knowledge. In fact, the design of Γ_τ reflects the prior about the shape, the expectation of the metric τ related to the shape and thus *what we know* about the shape.

3.3.1 A unique energy

Consider the sum term $\sum_{i=1}^{n_f} \gamma_i \mathcal{F}_i + \sum_{j=1}^{n_g} \lambda_j \mathcal{G}_j$ of formula 3.12, and suppose that all the probability density functions Γ_τ originating $\mathcal{F}_i$ and $\mathcal{G}_j$ are uncorrelated Gaussian ($\Gamma_\tau = \mathcal{N}_\tau(\mu_\tau, \Sigma_\tau)$, where μ_τ and Σ_τ are respectively the expected value and the covariance

matrix). The distribution

$$\Gamma = \mathcal{N}\left(\bar{\mu}, \bar{\Sigma}\right) = \mathcal{N}\left(\begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_i \\ \vdots \\ \mu_{n_f+n_g} \end{bmatrix}, \begin{bmatrix} \Sigma_1 & 0 & 0 & \cdots & & \\ 0 & \Sigma_2 & 0 & \cdots & & \\ 0 & & \ddots & & & \\ \vdots & & & \Sigma_i & & \\ & & & & \ddots & \\ & & & & & \Sigma_{n_f+n_g} \end{bmatrix}\right) \tag{3.14}$$

is a multivariate Gaussian distribution "grouping" all of them. Note that

$$\sum_{i=1}^{n_f} \gamma_i \mathcal{F}_i + \sum_{j=1}^{n_g} \lambda_j \mathcal{G}_j \approx \mathcal{N}\left(\bar{\mu}, \bar{\Sigma}\right) \tag{3.15}$$

especially in the sense of formula 3.1, being the result an argument minima. Thus formulas 3.12 and 3.10 can be restated as:

$$\begin{aligned} \int_{\partial C} \alpha(s) \left|\frac{\partial \mathcal{C}}{\partial s}\right|^2 + \beta(s) \left|\frac{\partial^2 \mathcal{C}}{\partial s^2}\right|^2 + \mathcal{D}\left[I\right] ds + \sum_{i=1}^{n_f} \gamma_i \mathcal{F}_i + \sum_{j=1}^{n_g} \lambda_j \mathcal{G}_j \approx \\ \int_{\partial C} \alpha(s) \left|\frac{\partial \mathcal{C}}{\partial s}\right|^2 + \beta(s) \left|\frac{\partial^2 \mathcal{C}}{\partial s^2}\right|^2 + \mathcal{D}\left[I\right] ds + n_f n_g \cdot \mathcal{N}\left(\bar{\mu}, \bar{\Sigma}\right) \end{aligned} \tag{3.16}$$

and

$$\mathcal{E}_g(\mathcal{C}) = \mathcal{S}_g(\mathcal{C}) + \mathcal{P}_g(\mathcal{C}) \approx \mathcal{S}(\mathcal{C}) + \mathcal{P}(\mathcal{C}) + n_f n_g \cdot \mathcal{N}\left(\bar{\mu}, \bar{\Sigma}\right) \tag{3.17}$$

Taking one further step, one may remove the assumption of independent measures:

$$\Gamma = \mathcal{N}\left(\bar{\mu}, \bar{\Sigma}\right) = \mathcal{N}\left(\begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_i \\ \vdots \\ \mu_{n_f+n_g} \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \Sigma_{13} & \cdots & & \\ \Sigma_{21} & \Sigma_{22} & \Sigma_{23} & \cdots & & \\ \Sigma_{31} & & \ddots & & & \\ \vdots & & & \Sigma_i & & \\ & & & & \ddots & \\ & & & & & \Sigma_{n_f+n_g, n_f+n_g} \end{bmatrix}\right) \tag{3.18}$$

This is especially true for measures that exhibits a high correlation, such as area and perimeter, inside/outside color, central moments.

3.3.2 Color clustering

Color clustering is the problem of partitioning an image I in areas of uniform color. A good clusterization divides I in few regions of almost uniform color [31]. The number of classes (or *clusters*) may be either defined a priori either dynamically adjusted

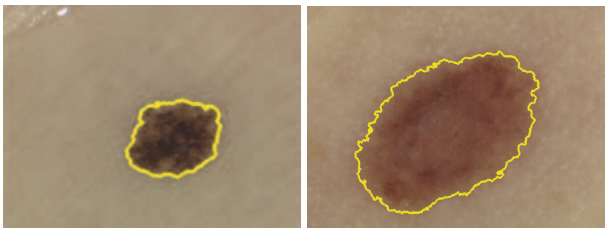


Figure 3.2: **Color segmentation of dermoscopic images** *Modified C-fuzzy provides a good segmentation for dermoscopic images. Two melanomas are here detected*

according to the complexity of the image. Figure 3.2 shows a typical output on dermoscopic images of melanocytic lesions. The result is a number of shapes, each of those defined by a group of adjacent pixels. Segmentation again involves equation 3.2. The slight difference is that the minimization is implicitly carried on a number of shapes (=cluster) rather than a unique one. Even if the color clustering minimization usually does not appears in analytical forms like equation 3.5 or 3.11, it is indeed just a different way to express the term $\mathcal{G}_j$.

The modified fuzzy c-mean algorithm we developed is an example. The algorithm computes the Principal Components of the image, using the Karhunen-Love transform, and then the 2-dimensional histogram $h(I)$ associated with the two components of largest variance. Given the number of clusters, the following two (recursive) equations optimally cluster the color space:

$$U_{k,\mathbf{x}} = \frac{1}{d(\mathbf{x},\mathbf{c}_k)^{b_1}} \tag{3.19}$$

$$\forall k: \quad \arg\min_{\mathbf{c}_k} \; I_k = \int_{h(I)} (U_{k,\mathbf{x}})^{b_2} |\mathbf{x},\mathbf{c}_k|^{b_3} h(\mathbf{x})^{b_4} d\mathbf{x} \tag{3.20}$$

where $\mathbf{x}$ is a point in the 2d color space, $\mathbf{c}_k$ is the center of cluster k in the color space, $U_{k,\mathbf{x}}$ refers to the fuzzy membership of $\mathbf{x}$ to cluster k, $|\cdot|$ is the Euclidean distance and b_1,b_2,b_3 and b_4 are scalar values obtained through learning process. The histogram clustering is then mapped back to the original image and a morphological postprocess removes the smallest areas.

Bringing back to equation 3.12, in the fuzzy c-mean we got $\alpha = 0$,$\beta = 0$, $\gamma_i = 0$, $\mathcal{D}[I] = 0$ and $n_g = 1$. In a nutshell, the unique (image) energy term $\mathcal{G}_1$ is expressed by equation 3.20, and there are no constraints on the shape model nor on the image derivatives.

3.4 Case studies

3.4.1 Miocardial cells

Miocardial cells balance the internal forces and external activity, yielding a rest shape that is a tradeoff between internal spring forces and external stretching activity. Cells

evolve in time while continuously changing the shape appearance by protruding appendages that adhere to the culture surface to force and help the deformation evolution ($1D$-side of the shape characterization). In this sense, the smoothness of the boundary profile counteracts this behavior: conversely, by acting on a term allowing the boundary curvature κ to assume negative values for some time windows, a spiky appearance is enforced.

$$\mathcal{S}_{spike}(C) = \int_S \kappa(s)ds \qquad \kappa(s) = \frac{\partial^2 C(s)}{\partial s^2} \tag{3.21}$$

Following formula 3.13, we define a spikeness-related probability density function as

$$\Gamma_{spike} = \frac{min(\mathcal{S}_{spike}, \mathcal{S}_{max})}{\mathcal{N}} \tag{3.22}$$

and a related energy term

$$\mathcal{E}_{spike} = -\Gamma_{spike} \tag{3.23}$$

where $\mathcal{N}$ is a normalizing scalar value to make the integral of 3.21 converge to 1. Γ_{spike} is thus a piece-wise linear function with upper bound $\mathcal{S}_{max}$. For our practical purposes, $\mathcal{S}_{max}$ was choosen to be an unreachable high value.

Cells viewed from a microscope often show a bright halo, due to the scattering properties of the living material enhanced by the thinness of the cells at the border. At the same time, protein appendices absorb light, resulting in black spots near the end of the fiber. These optical phenomena are very common in back-lit organic materials and can be used to attract the contour in absence of strong edges. This information is related to the inside cell texture (f_w defined inside X) and neighborhood features (f_b defined outside X, ΔX), and these data related contributions ($2D$ shape descriptors) are referred as

$$\begin{aligned} \mathcal{P}_W(C) &= \int_X [I(\omega) - f_w(\omega)]^2 \, d\omega \qquad f_w(\omega) = \text{inner texture prototype} \\ \mathcal{P}_b(C) &= \int_{\Delta X} [I(\omega) - f_b(\omega)]^2 \, d\omega \qquad f_b(\omega) = \text{neighborhood prototype} \end{aligned} \tag{3.24}$$

Using again 3.13, we build two probability Gaussian density functions

$$\Gamma_w = \frac{1}{\sigma_w\sqrt{2\pi}} \cdot e^{-\frac{\mathcal{P}_W(C)^2}{2\sigma_w^2}} \tag{3.25}$$

$$\Gamma_{f_w} = \frac{1}{\sigma_{f_w}\sqrt{2\pi}} \cdot \cdot e^{-\frac{\mathcal{P}_b(C)^2}{2\sigma_{f_w}}} \tag{3.26}$$

and hence two energy terms

$$\mathcal{E}_w = -\Gamma_w \tag{3.27}$$

$$\mathcal{E}_b = -\Gamma_b \tag{3.28}$$

where σ_w and σ_{f_w} are suitable scalar values. For our experiments we ofter chose $\sigma_w = \sigma_{f_w} = 1$. Fig. 3.5 presents the results of the application of the proposed model

in some frames from a video sequence of a cell deformation. The standard Active Contour formulation is not able to correctly detect the cell shape: errors accumulate over frames and end up with a general failure of the tracking algorithm. Nonetheless, even in the first frames of the sequence, the Active Contour fails in capturing all the fine details, such as protein spikes and ridges, yielding erroneous statistics. For the definition of the energetic terms, we used a number of metrics (see figure 3.3 and 3.4, and chapter 8).

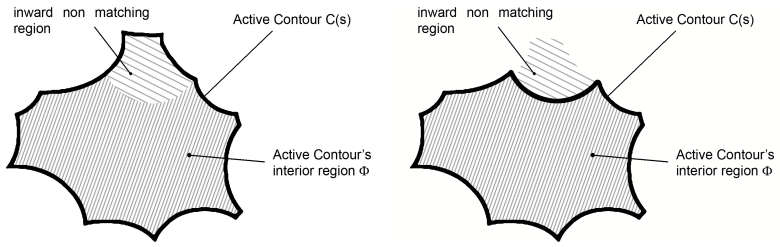


Figure 3.3: **Texture difference** *Generalized Active Contour minimizes $\mathcal{E}_W$, in the example leading to a contraction of the shape, pulling-out the different textured region. Referring to figure 3.5, $\mathcal{E}_W$ forces the bright halo surrouding the cell to be left out*

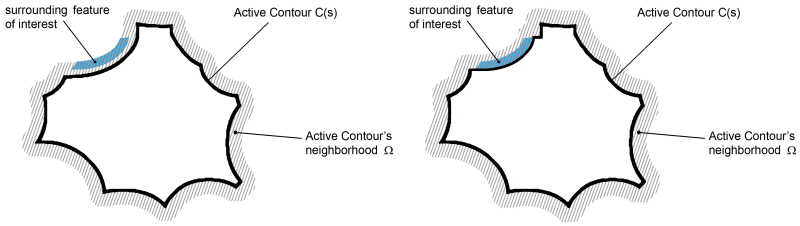


Figure 3.4: **Neighbor texture difference** *$\mathcal{C}(s)$ tries to minimize the neighborhood energy $\mathcal{E}_{f_W}$. In the example, $f(\omega)$ specify that the surrounding color has to be "blue", and thus the Snake is attracted by the blue spot. Referring to figure 3.5, $\mathcal{E}_{f_W}$ forces the dark protein appendices to be included*

3.4.2 Cardial Patch

The patch alternates periods of inactivity to sudden movements. To understand and measure the temporal sampling frequency is crucial: even though the videoswas clear and color-contrasted, we found that the frame rate was too low.

We build a color-related energy term, able to capture the reddish back-scattering of the patch. The following is a measure of the red contrast of the shape

$$\mathsf{R}(\mathcal{C}) = \frac{\int_X I_r dx}{\int_X dx} \tag{3.29}$$

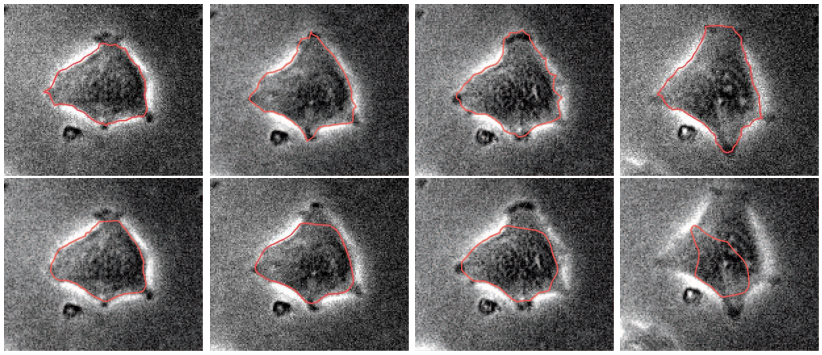


Figure 3.5: **Miocardial cell shape detection** *The first row is obtained using the proposed Active Contour model, while the second row is produced with the original formulation. The model uses as* f_w *a constant zero-function (the black color) and as* f_b *a constant* 255 *function (the white color)*

where I_r indicates the red channel of the color image I. Following the example traced out in equation 3.22 we build a piecewise linear probability density function:

$$\Gamma_{red} = \frac{min\left(\mathsf{R}, \mathsf{R}_{max}\right)}{\mathsf{N}} \tag{3.30}$$

where again N is a normalizing value and R_{max} is an upper bound value for R. $\mathcal{E}_{red}$ is again build as simply as

$$\mathcal{E}_{red} = -\Gamma_{red} \tag{3.31}$$

Figure 3.6 shows the framework applied on the “cardiac patch sequence”

Figure 3.6: **Cardiac pulsating patch** *The cardiac patch sequence presents abrupt movements and sudden changes of position and size of the tissue. The* $\mathcal{E}_{red}$ *term attracts the contour towards a good fit even when the movements is large, in a unique pseudo time integration*

3.4.3 mbari video

For the "sea flower" sequence [1] we used a technique similar to the cardial tissue, exploiting the lightness difference in the blue color channel. The energy term E_b obliges the Active Contour to capture a shape with high blue contrast:

$$\mathtt{B}(\mathcal{C}) = \frac{\int_X I_b dx}{\int_X dx} \tag{3.32}$$

where I_b indicates the blue channel of I,

$$\Gamma_{blue} = \frac{min\,(\mathtt{B}, \mathtt{B}_{max})}{\mathtt{N}} \tag{3.33}$$

and $\mathtt{B}_{max}$ is an upper bound value for $\mathtt{B}$. $\mathcal{E}_{blue}$ is again build as simply as

$$\mathcal{E}_{blue} = -\Gamma_{blue} \tag{3.34}$$

Figure 3.7 shows results from a video sequence of 350 frames.

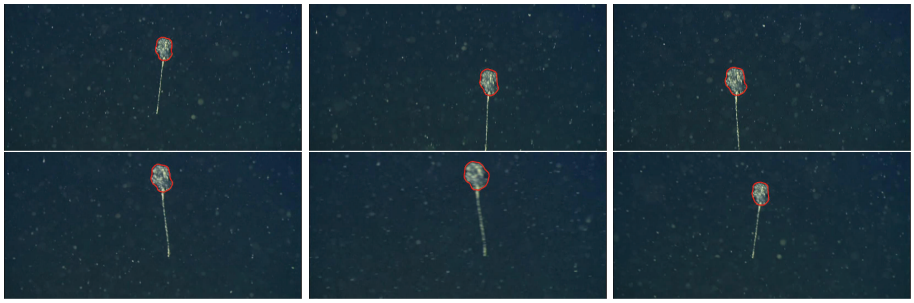


Figure 3.7: **Sea Flower image sequence** *Active Contours are unreliable on blurry images such as underwater shots, where camera movements and the diffusion of the light make edges dull. Generalized Active Contour provides a better shape detection tool, in this case easily embedding the color information*

[1] courtesy of Dr. D.Cline, www.mbari.org

3.5 Experiments

3.5.1 Miocardial cells

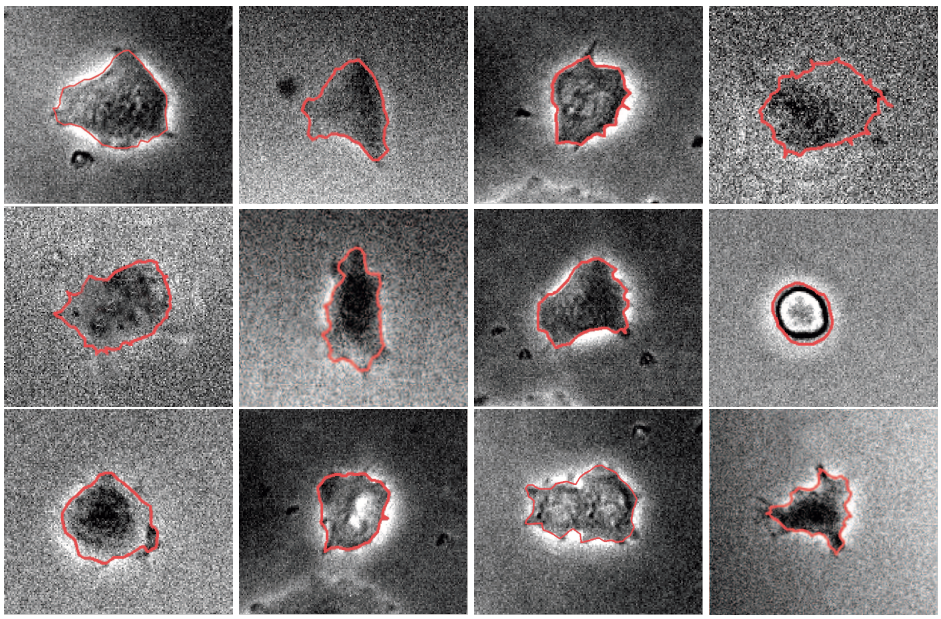


Figure 3.8: **Miocardial cell single shape detection**. *Generalized Active Contours are a reliable tool to detect shapes. The figure shows 12 still images and the detected shapes (red lines). Each shape was obtained using the same energy terms, without any parameter tuning. Images are here more contrasted than those actually used in the experiments to make them more visible in this work*

The section displays 12 frames taken from the 10 miocardial cells image sequences. Figure 3.8 shows Photoshop contrasted frames, artificially altered for the purpose of clarity for this work (see figure 5.7 to have an example of the real data). Generalized Active Contours proved to be an effective tool to detect shapes, with no or little tuning required: each shape was in fact obtained using the same energy terms.

3.5.2 Cardial patch

The section presents the results of the Generalized Active Contours applied to the two cardial patch sequences (see section 3.4.2). The video sequences are characterized by a similar behavior of the tissue. Figure 3.9 displays the user interface we developed to support non technicians.

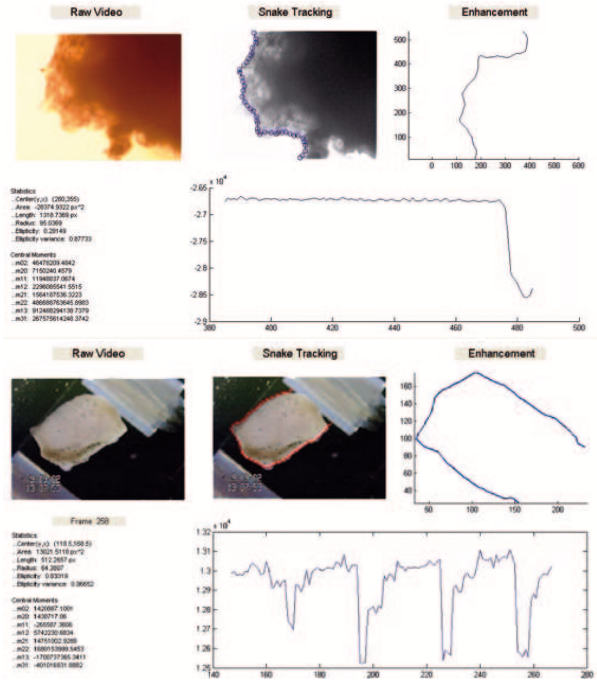


Figure 3.9: **Cardiac pulsating patch** *The cardiac patch sequence presents abrupt movements and sudden changes of position and size of the tissue. The $\mathcal{E}_{red}$ term attracts the contour towards a good fit even when the movements are large, in only simulation step*

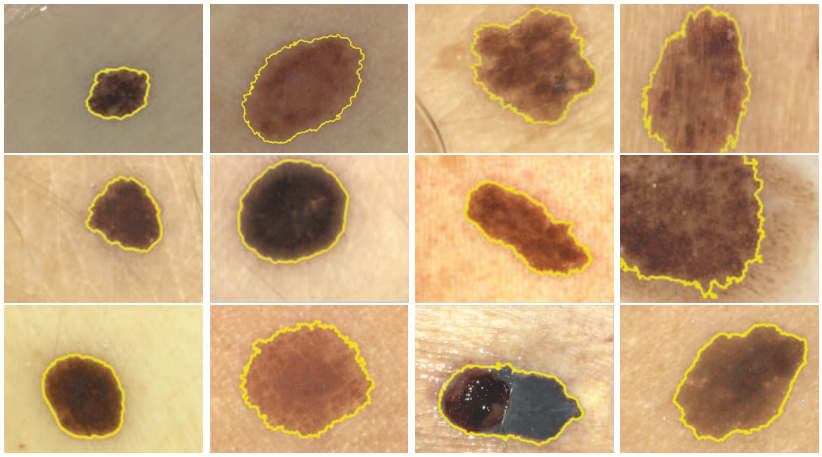


Figure 3.10: **Color segmentation of dermoscopic images**

3.5.3 Melanoma

In figure 3.10 we presents the results of the c-fuzzy color clustering algorithm on 12 dermoscopic images of melanocytic lesions.

4

Reticular Shape detection

This chapter deals with the detection of reticular shapes. Section 4.1 introduces the problem and provides the theoretical background while section 4.2 presents a novel approach based on Random Walk Agents. Section 4.3 shows processes results to remove false positive detections. Section 4.4 closes the chapter showing results obtained on the drosophila epithelium and the corneal fundus.

4.1 Reticular shape detection

Reticular shapes arise in a number of situation in nature and human design (see figure 4.1). The detection of such structures again follows equations 3.1 and 3.2 of chapter

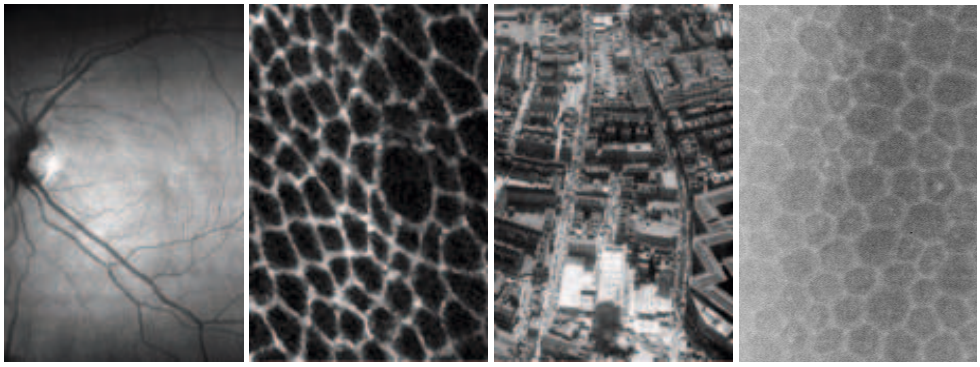


Figure 4.1: **Examples of reticular shapes**. *Reticular shapes arise commonly in nature as well in the industrial design. Tightly packed structures such as cells in the ephitelium or blood vessels originates such shapes. From left to right: retina blood vessels, drosophila melanogaster epithelium, aerial street view, corneal fundus*

3.

Given the high degree of freedom of reticular structures, ordinary approaches to

minimization such as the ones presented in chapter 3 and 5 do not yield satisfactory results. From one side, the computation complexity is higher than the single shape detection; from the other side, the design of the energetic term $\mathcal{E}$ is not well understood.

Next section presents a Random Walk approach, a tool to emulate the human vision when capturing reticular structure. The idea is to flood the image I with Random Walk agents, each of them finding a path in the digital frame. Path are created by locally minimazing an energy term $\hat{\mathcal{E}}$, being an analytical approximation of $\mathcal{E}$. In general, $\mathcal{E}$ is not known or difficult to obtain, whereas $\hat{\mathcal{E}}$ isn't. In a nutshell, the Random Walk Agents approach shifts the problem from the (difficult) one of designing $\mathcal{E}$ to the easier one of designing $\hat{\mathcal{E}}$.

4.2 The Random Walk Agents

A Random Walk Agent is an agent $\mathcal{A}$ that travels the digital frame, tracking paths over its passage [4], [29]. In its basic incarnation, an agent is a discrete time system with an internal walk model and an external input $\hat{\mathcal{E}}$:

$$\mathbf{p}(t+1) = \mathbf{p}(t) + k \cdot g\left(\hat{\mathcal{E}}(t)\right), \tag{4.1}$$

where

$$\mathbf{p}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} \in \mathbb{L}, \tag{4.2}$$

is the current point position on the domain $\mathbb{L} \subset \mathbb{R}^2$ of the image frame and $g(\cdot)$ is a heading direction function, depending on $\hat{\mathcal{E}}$, position, direction and speed of the motion. k is a constant scalar value, defining the advancement step size.

The following is the observation equation:

$$\mathbf{y}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{p}(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \theta(t). \tag{4.3}$$

Note that here the variable t refers to the agent pseudo-time, not to a collection of time-related digital images such as in chapter 5. Note also that equation 4.1, keeping things very didactic, uses the simplest possible walk model. More complex models are of course viable.

The role of $\hat{\mathcal{E}}$ is to drive the agent $\mathcal{A}$ towards those locations of the image characterized by the presence of the reticular structure. We'll discuss more on the design of $\hat{\mathcal{E}}$ in the next section.

Several instances of the explorer agent, $\{\mathcal{A}_i, i = 1, \ldots, N_{\mathcal{A}}\}$, are concurrently present in the field of vision, and each generates at any time more than one direction that is viable to advance the exploration. More precisely, for each pair $\{x,y\}_i$ the set of m_i possible directions $\{\theta_{i,j}, j = 1, \ldots, m_i\}$– originating $(m_i - 1)$ new agents–are collected in the vector $\Theta_i \in \mathbb{R}^{m_i}$.

Globally, the observations $\mathbf{y}(t)$ of all the explorer agents $\mathcal{A}_i$ yield a graph model $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ (being $\mathcal{N}$ and $\mathcal{E}$ nodes and edges, respectively), where each node $n_i \in \mathcal{N}$ is characterized by the state $\{x, y, \Theta\}$ of the locations visited by one of the $\mathcal{A}_i$, and the edges keep track of the path traveled by each agent (figure 4.2). This graph provides a good *representation* of the retrieved structure. The level of detail embedded in the description can be augmented either with a more refined algorithmic procedure or with a finer discretization of the input data.

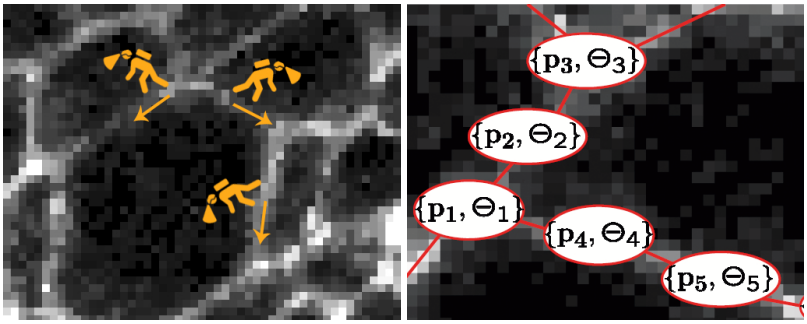


Figure 4.2: **The Random Walk approach**. *Several agents (left, in yellow) explores the frame. The set of the paths of all the agents is the recovered reticular structure. On the right, a possible graph representation models, where nodes corresponds to the locations* $[x(t), y(t)]$ *and edges connect two adjacent nodes*

4.2.1 $\hat{\mathcal{E}}$ design

Consider an agent $\mathcal{A}$ in a location $[x(t), y(t)]$ of the frame. The purpose of $g(\hat{\mathcal{E}})$ is to provide a set of directions $\Theta = \{\theta_1, \ldots, \theta_{N_\mathcal{A}}\}$ for advancing such that $[x(t+1), y(t+1)]$ is still a location on the reticular structure. With reference to a pictorial interpretation, the aim is thus to explore the surroundings of the current position and intuitively move from a good location to another good location.

$\hat{\mathcal{E}} : [0, 2\pi) \rightarrow \mathbb{R}$ gathers local information. For each sector Ω_i centered on the current position $\mathbf{p}$, the image texture is compared with a reference one, according to:

$$\hat{\mathcal{E}}(\theta_i) = \frac{\int_{\Omega_i} \sqrt{(I(\omega) - I_{ref}(\omega))^2} d\omega}{\int_{\Omega_i} d\omega} \tag{4.4}$$

A natural choice is to build circular sectors Ω_i such as in figure 4.3, on the right. Sectors of different shapes are still possible: for a number of the problems of interest rectangular sectors also proved to be effective (figure 4.3, right).

The set of the heading directions is given by:

$$\Theta = \{\theta_1, \ldots, \theta_{N_\mathcal{A}}\} = \arg_\theta \frac{\partial \hat{\mathcal{E}}}{\partial \theta} = 0 \text{ and } \hat{\mathcal{E}} < threshold \tag{4.5}$$

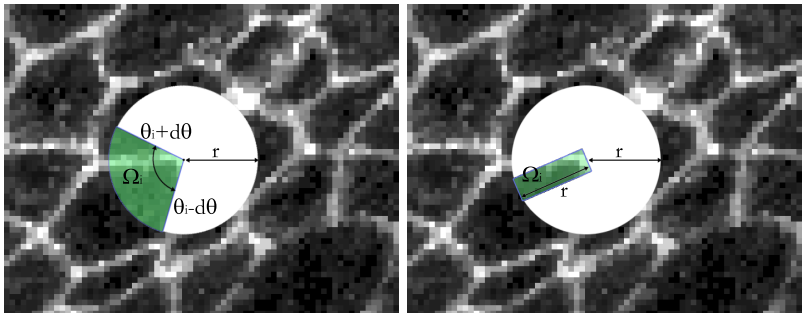


Figure 4.3: **Sectors of different shape**. *A template function built on circular sectors is a natural choice, since it mimics the field of view of human beings (left). Other shapes are also possible, depending on the problem: in the drosophila case, a rectangular shape efficiently serves the purpose (right)*

where the "threshold" cuts out all the direction with a high energy cost. $\frac{\hat{\mathcal{E}}}{\partial\theta} = 0$ enforces that the point has to be a minimum.

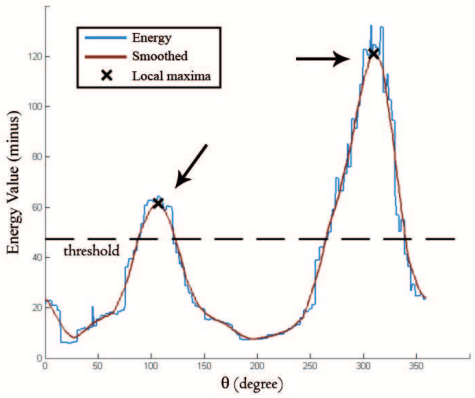


Figure 4.4: $\hat{\mathcal{E}}$ **function**. *Local minima of the energy function (here displayed as .$\hat{\mathcal{E}}$) correspond to heading direction for advancing. A smoothed version, filtering small random disturbances, is more reliable for noisy images*

The heading directions θ_i are the inputs of equation 4.1, thus generating a corresponding number of agents $\{\mathcal{A}_i, i = 1, \ldots, N_{\mathcal{A}}\}$.

Let us first observe that small loops have to be avoided. Small loops are frequent in uniform areas of the image, where the energy function $\hat{\mathcal{E}}(\theta)$ is not prone to differentiate between redundant directions. In these situations, a direction is as likely as any other, and the choice is driven mainly by noise and randomness. This happens because $|\Theta| >> 1$ and $\theta_i \approx \theta_{i+1}\ \forall\ \theta_i \in \Theta$. In other words, all the values of $\theta_i \in \Theta$ are very similar and these directions spans uniformly the surrounding environment, so there is no particular reason to pick one direction over another. This limitation is due to the local nature of the brightness function $L(\theta)$. Nonetheless this is still a desirable

feature because it bounds the algorithmic complexity and it respects the fundamental *walking idea* behind the algorithm.

In order to avoid the aforementioned small loops, after extracting a point from the queue, it is tested against the creation of loops in the graph. Large loops are accepted because they correspond to actual closed paths (for instance the perimeter of a cell), while small loops are disregarded (figure 4.5).

A joining procedure is also required to fill small gaps and close large loops: if two agents $\mathcal{A}_i$ and $\mathcal{A}_j$ end up close to each other such that $|\{x,y\}_i - \{x,y\}_j| < \varepsilon$, where ε is a threshold, they are joined together.

The algorithm also marks as *boundary nodes* those nodes close to the boundary of the digital image: The expansion process ends there (figure 4.5).

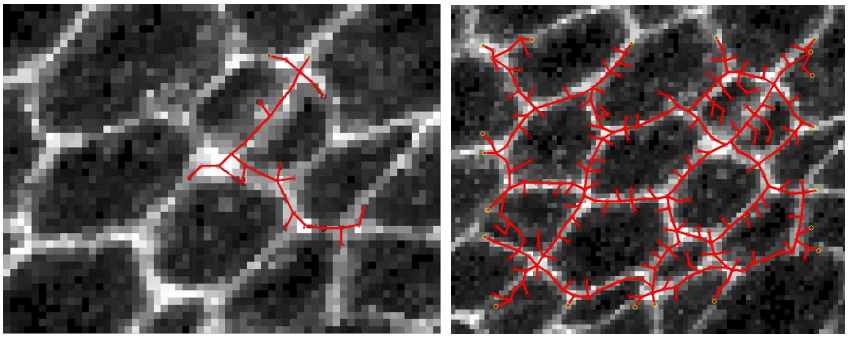


Figure 4.5: **Small loops creation and dead branches**. *The early closure of the path, and thus the creation of small loops, is a frequent issue (left). This is due to the locality of the energy function. These loops are often false positives. Dead branches (right) span over the interior of the cells*

4.3 Post processing

Random Walk Agents generate a structure spanning over the original reticular shape, but the expansion process may create dead-branches. These branches are characterized by extremal non-border nodes of degree 1. A cleaning post-process takes the graph as its input, and iteratively removes all such points. The procedure is iterative and stops when no more removal occurs. A snapping procedure then joins the following types of points:

- a pair of neighboring nodes of degree $d > 2$
- a node of degree $d > 2$ and a neighbor border node
- a pair of neighbor border nodes

Again, the procedure is iterative and stops when no more fusions takes place.

4.4 Case studies

4.4.1 Drosophila epithelium

Drosophila reticular structure presents bright patterns of light superimposed to a dark background. The brightness is due to the protein markers, that adhere to the cellular boundaries and results in white when viewed by means of a microscope.

Thus, when viewed from atop, the entire structure is "bright" and "white": somehow, the energy terms have capture this. As discussed in the introduction, the design of $\hat{\mathcal{E}}$ should reflects locally the global properties of the shape to detect. In particular, we applied equation 4.4 and chose as I_{ref} the constant-value function 1 (the color code for white in RGB images):

$$\hat{\mathcal{E}}(\theta_i) = \frac{\int_{\Omega_i} \sqrt{(I(\omega)-1)^2}\, d\omega}{\int_{\Omega_i} d\omega} \tag{4.6}$$

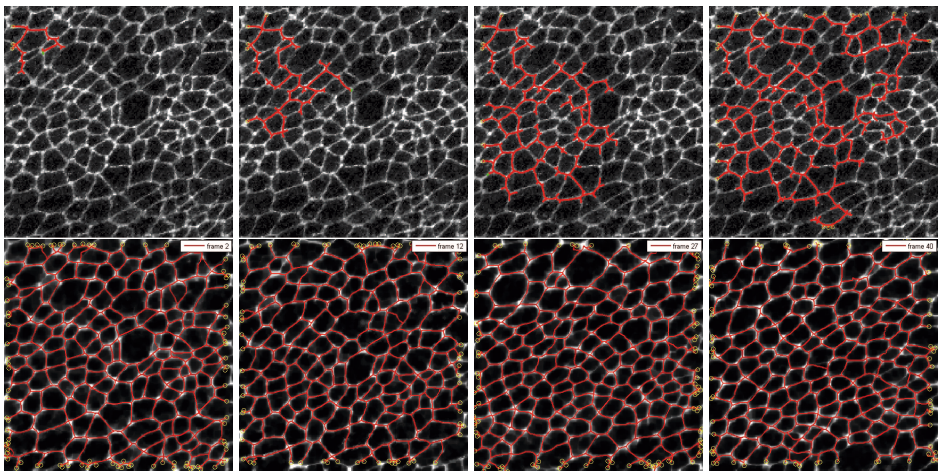


Figure 4.6: **Detection of the reticular shape on the drosophila "wing" epithelium**. *Cells' boundaries are highlight in white using a marker protein. The Random Walk Agents run all over the frame in the attempt of minimizing equation 4.6. The bottom row shows results on several images (frame* 2,12,27,40 *from the "wing" sequence)*

Figure 4.6 shows results applied to a number of images of the drosophila wing epithelium. On an average of 80 images, the false positive error rate (detection of an unexisting edge) is 5% and the false negative (missing an existing edge) is 6%. The ground truth was provided by an unbiased human observer. Each frame was considered as unique and processed with the Random Walk Agents algorithm, we did not exploit the temporal coherence as in section 5.4.2. The top row shows the typical flooding behavior of the walkers. The 4 images 4.6 were obtained halting the algorithm respectively at the beginning, at $1/4$, $1/2$ and almost $3/4$ of the total computation.

Analyzing the results we noted some blank areas not covered by the walker. This is due to the absence of the protein markers on the edges of the cells, resulting in poorly contrasted regions

Figure 4.7 shows the result from the “notum” still frame. Here the protein marker is different from the one used in figure 4.7, as well the illumination, the number of cells and the resolution of the image.

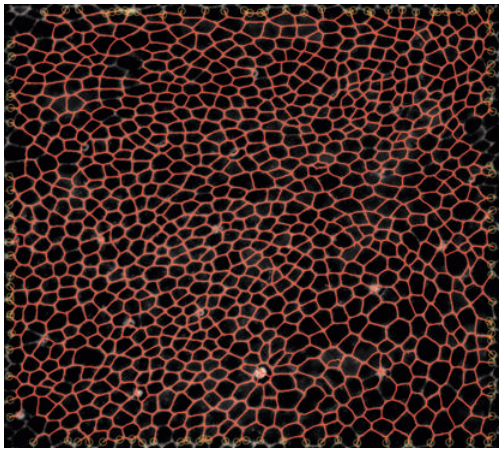


Figure 4.7: **Detection of the reticular shape on the drosophila “notum” epithelium**. *The very high resolution frame shows the result of the Random Walk approach on the “notum” frame*

4.4.2 Corneal fundus

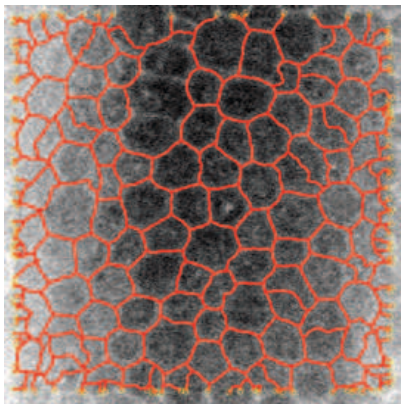


Figure 4.8: **Preliminary results from a corneal fundus image**. *The image shows preliminary results from the corneal fundus. Errors are mainly due to the background illumination. An image preprocessing step is here needed to adjust difference of lightning*

5

Dynamic Single Shape detection

The chapter extends the single shape detection framework developed in chapter 3 to a sequence of images. Section 5.1 states the basis of the problem and provides a theoretical solution. Section 5.2 exploits the images temporal coherence to design a faster and more robust detection, while section 5.3 extends the approach to the Generalized Active Contours (developed in 3.3). Finally, section 5.4 shows results in a number of case studies.

5.1 Tracking - Dynamic Single shape detection

The natural extention of the shape detection problem as seen in chapter 3 is to consider a (time) sequence of images $\{I_1, \dots I_t, \dots, I_N\}$. We call this problem *dynamic* single shape detection, referring to the evolution of the shape over time.

The goal is to detect the object of interest X in each image I_t, obtaining a sequence of shapes $\{\Phi_1(X), \dots \Phi_t(X), \dots, \Phi_N(X)\}$. The common approach is to apply equation 3.2 at each time step:

$$\begin{aligned}
\Phi_1 &= \arg\min_{\Phi_c \in S} \mathcal{E}_1(\{I_1, \dots, I_N\}, \Phi_c) \\
&\dots \\
\Phi_t &= \arg\min_{\Phi_c \in S} \mathcal{E}_t(\{I_1, \dots, I_N\}, \Phi_c) \\
&\dots \\
\Phi_N &= \arg\min_{\Phi_c \in S} \mathcal{E}_N(\{I_1, \dots, I_N\}, \Phi_c)
\end{aligned} \tag{5.1}$$

It is a common assumption to consider $\mathcal{E}_t$ dependent uniquely on the current time. Thus $\mathcal{E}_t(\{I_1, \dots, I_N\}) = \mathcal{E}_t(I_t))$ and $\mathcal{E}_1 = \mathcal{E}_2 = \dots = \mathcal{E}_N$:

$$\begin{aligned}
\Phi_1 &= \arg\min_{\Phi_c \in S} \mathcal{E}(I_1, \Phi_c) \\
&\cdots \\
\Phi_t &= \arg\min_{\Phi_c \in S} \mathcal{E}(I_t, \Phi_c) \\
&\cdots \\
\Phi_N &= \arg\min_{\Phi_c \in S} \mathcal{E}(I_N, \Phi_c)
\end{aligned} \tag{5.2}$$

5.2 An incremental approach

The lack of temporal coherence is the main disadvantage of the approaches 5.1 and 5.2. Simple stated, temporal coherence is the property of an image frame I_t to be similar to its neighbor I_{t+1}. Formally:

$$I_{t+1} = I_t + \Delta_t^{t+1} \tag{5.3}$$

where Δ_t^{t+1} is a function of the same domain of I_t and Δ measures the difference between two consecutive frames. The temporal coherence is high when $|\Delta| \approx 0$ and low when $|\Delta| >> 0$, according to a norm $|\cdot|$. Equation 5.3 suggests that one can compute Φ_{t+1} as an adjustment of Φ_t

$$\Phi_{t+1} = \Phi_t + \delta_t^{t+1} \tag{5.4}$$

where δ_t^{t+1} is a "morphing" of Φ_t into Φ_{t+1}. The role of δ is analogous to Δ, save for δ is here an abstraction whose analytical form depends on the shape representation model.

To clarify the idea, we anticipate concepts of section 5.3, considering the Generalized Active Contours presented in section 3.3. Here δ is a displacement vector field applied pointwise to $\mathcal{C}(s)$. δ, evaluated in position $\mathcal{C}(s)$, is the displacement vector mapping $\mathcal{C}_t(s)$ into $\mathcal{C}_{t+1}(s)$

$$\mathcal{C}_{t+1}(s) = \mathcal{C}_t(s) + \delta_t^{t+1}(\mathcal{C}_t(s)) \tag{5.5}$$

Following equation 5.2, the optimum δ_t^{t+1} satisfies

$$\Phi_{t+1} = \Phi_t + \delta_t^{t+1} = \arg\min_{\Phi_c \in S} \mathcal{E}(I_{t+1}, \Phi_c) \tag{5.6}$$

and hence the chain solution for 5.2 is

$$\begin{aligned}
\Phi_1 &= \arg\min_{\Phi_c \in S} \mathcal{E}(I_1, \Phi_c) \\
\Phi_2 &= \Phi_1 + \delta_1^2 \\
&\cdots \\
\Phi_{t+1} &= \Phi_t + \delta_t^{t+1} \\
&\cdots \\
\Phi_N &= \Phi_{N-1} + \delta_{N-1}^N
\end{aligned} \tag{5.7}$$

5.3 Active Contour incremental approach

Velocity or Optical Flow $\mathcal{F}$ is a concept of computer vision literature [6, 3, 28]. Given two images I_t and I_{t+1}, the purpose of the optical flow is to assign each pixel in I_t to a position into I_{t+1}, not necessarly a unique pixel or a location inside the frame. Commonly, optical flow algorithms output a $\mathbb{R}^2 \to \mathbb{R}^2$ function, a vector field whose elements $\mathcal{F}(x,y) = (V_x, V_y)$ describe the displacement of pixels (x,y) of image I_t into I_{t+1}. Roughly speaking:

$$I_t(x,y) = I_{t+1}(x+v_x, y+v_y) \tag{5.8}$$

In the following, and only for this section, we keep notation consistent with the Taylor expansion series using $I_t(x,y) = I(x,y,t)$.

Considering small displacements and small time steps, equation 5.8 becomes

$$I(x,y,t) = I(x+\delta x, y+\delta y, t+\delta t) \tag{5.9}$$

which is known as the *image constraint equation*. Assuming the movement to be small, the image constraint at $I(x,y,t)$ through Taylor series can be rewritten as

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + h.o.t \tag{5.10}$$

where h.o.t. means higher order terms, usually neglected. From these equations it follows that:

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0 \tag{5.11}$$

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} = 0 \tag{5.12}$$

which results in

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0 \tag{5.13}$$

or more commonly

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y = -\frac{\partial I}{\partial t} \tag{5.14}$$

5.14 is an equation in two unknowns and cannot be solved as such. This is known as the *aperture problem*. To find the optical flow another set of equations is needed, given by some additional constraint.

5.3.1 J-maps

This section presents a method to compute the optical flow efficiently. The optical flow is computed only on specific locations, namely $\mathcal{C}(s)$.

For a given s, Consider $\mathcal{N}_t(\mathcal{C}_t)$ a circular portion of a certain radius of I_t, centered on $\mathcal{C}_t(s)$. Recalling section 1.3, $\mathcal{C}_t(s)$ is a R^2 point. Consider now $\mathcal{N}_{t+1}(\mathcal{C}_t + [\hat{v}_x, \hat{v}_y])$,

an analogous circular portion of I_{t+1} centered on $\mathcal{C}_t(s) + [\hat{v}_x, \hat{v}_y]$. $\mathcal{N}_t(\mathcal{C}_t) \subset I_t$ and $\mathcal{N}_{t+1}(\mathcal{C}_t + [\hat{v}_x, \hat{v}_y]) \subset I_{t+1}$.

We compute the optical flow $\mathcal{F}(\mathcal{C}(s))$ according to:

$$\mathcal{F}(\mathcal{C}(s)) = \mathcal{F}(x,y) = (v_x, v_y) = \arg\min_{\hat{v}_x, \hat{v}_y} \int_A \left| \mathcal{N}_t(\mathcal{C}_t) - \mathcal{N}_{t+1}(\mathcal{C}_t + [\hat{v}_x, \hat{v}_y]) \right| da \quad (5.15)$$

where A is a 2D domain. In a nutshell, we choose as $\mathcal{F}(\mathcal{C}(s))$ the vector $[v_x, v_y]$ that minimizes the difference between $\mathcal{N}_t(\mathcal{C}_t)$ and $\mathcal{N}_{t+1}(\mathcal{C}_t + [v_x, v_y])$.

On images with low coherence, the approach still holds but has to be revised. Equation 5.15 may in fact output a sub-optimal vector. This is especially true in presence of several minima of the same magnitude. Figure 5.1 shows two typical maps of the value of the integral in 5.15 according to $\hat{v}_x, \hat{v}_y$. High coherence images (left) generate a precisely located and unique minima, while low-coherence images (right) exhibit more than one good minima. We call these maps "J_s", or *J_s-maps*, where s indicates that they refer to the point $\mathcal{C}(s)$.

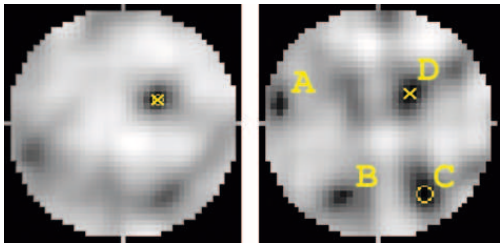


Figure 5.1: **J-maps**. *Highly coherent images I_t and I_{t+1} generates a J-map with a unique "good" minimum (left). The yellow circle indicates the minimum of the integral 5.15 (v_x and v_y), while the yellow cross the ground-truth provided by a human. Low coherent images (right) generates instead a J-map with many local minima of similar value (A,B,C,D). Here the global minima (yellow cross, D) does not correspond to the ground truth (circle, C)*

Given the situation depicted on figure 5.1, it's impossible to assess the position of the true minima. The idea here is not to consider a single J-map, rather a set of them, precisely a set of neighbor J-maps and to mutually correct them. The intuition holds true because the optical flow $\mathcal{F}(x,y)$ has to be smooth, $|\nabla \mathcal{F}| \approx 0$. Thus, neighbor J-maps will have similar values and similar minima. In other words, if we consider a set of close points $\{\mathcal{C}(s_1), \mathcal{C}(s_2), \ldots, \mathcal{C}(s_n)\}$ and apply the smoothness constraint, we get $\mathcal{F}(\mathcal{C}(s_1)) \approx \mathcal{F}(\mathcal{C}(s_2)) \approx \ldots \mathcal{F}(\mathcal{C}(s_n))$ and thus $J_{s_1} \approx J_{s_2} \approx \ldots \approx J_{s_n}$.

Figure 5.2 shows a set of neighbor J-maps (J-maps associated to neighbor points), with a yellow cross on the global minima. Intuitively, it is possible to detect the error on the right and to (somehow) correct it.

The J-maps correction is an iterative steps. For each image J_s we consider a set of neighbor images $\{J_{s_1}, J_{s_2} \ldots J_{s_n}\}$, and compute the global minima of each of them:

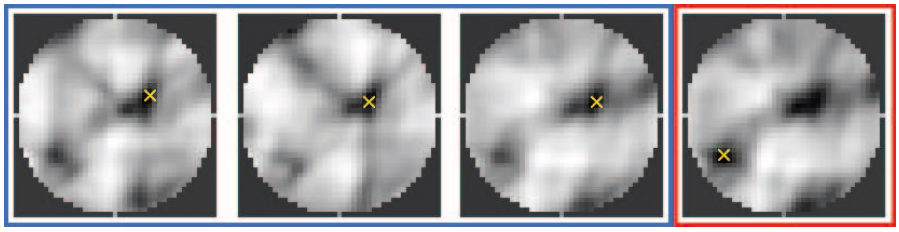


Figure 5.2: **Low coherent J-maps**. *J-maps from low coherent images show a number of local minima, and sometimes fail in detecting the ground truth. Nonetheless neighbor J-maps exhibits a similar appearance, and thus it's possible to cross check them and detect potention errors. For instance the image on the right shows a global minima not consistent with the global minima of the other J-maps - not consistent with the smoothness constraint*

$\{(v_{x1}, v_{y1}), (v_{x2}, v_{x2}), \ldots (v_{xn}, v_{xn})\}$. The average value

$$\bar{v_x} = \sum_{i=1}^{n} v_{xi} \qquad \bar{v_y} = \sum_{i=1}^{n} v_{yi} \tag{5.16}$$

and the covariance matrix

$$\Sigma_{xy} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{xy} & \sigma_{yy} \end{bmatrix} \tag{5.17}$$

indicates respectively the optical flow average direction in $\mathcal{C}(s)$ and the confidence of this measure.

We correct J_s according to

$$\bar{J_s} = J_s \cdot (1 - \mathcal{N}([\bar{v_x}, \bar{v_y}], \Sigma_{xy})) \tag{5.18}$$

where (N) is the Gaussian function. Image 5.3 shows equation 5.18 applied to the example of image 5.2. The image on the right is J_s, and the three remaining images on the left are $\{J_{s_1}, J_{s_2}, J_{s_3}\}$. After the correction step over all the images, the "new"

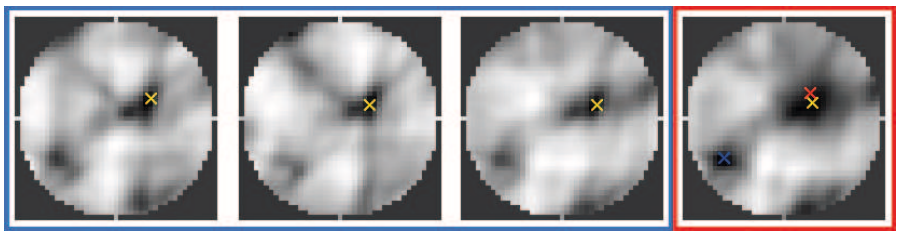


Figure 5.3: **Correction of the J-Maps**. *The J-map on the right, $\bar{J_s}$, is computed from J_s according to equation 5.18. The blu cross is the position of the old global minimum. The red cross is the center of the gaussian $(\bar{v_x}, \bar{v_y})$. The yellow cross is the location of the new global minimum*

$\bar{J}$ overwrite the "old" J. The entire procedure may be repeated, until a steady state is reached.

Figure 5.4 compare the morphing of a reticular structure computed respectively with the raw optical flow algorithm and the improved one. Figure 5.5 compare the

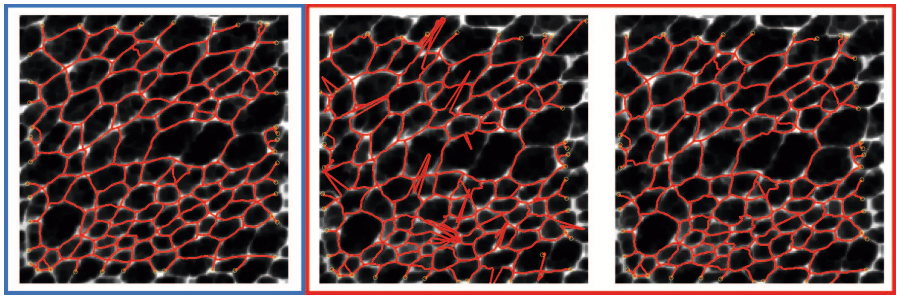


Figure 5.4: **Morphing of a reticular structure**. *The inexact computation of the optical flow can morph a shape (left, a reticular shape in red) into a wrong one (center). The improved optical flow algorithm guarantees instead optimal results (right). The images refer to a video sequence of the drosophila morphogenesis, captured by mean of a microscope. Red lines, over imposed on the original frames, show the cellular structure. The image on the left refers to frame 21, images on center and on the right refer to frame 22. See also images 5.5. Courtesy of Prof. Jeff Axelrod lab*

optical flows after and before refinement.

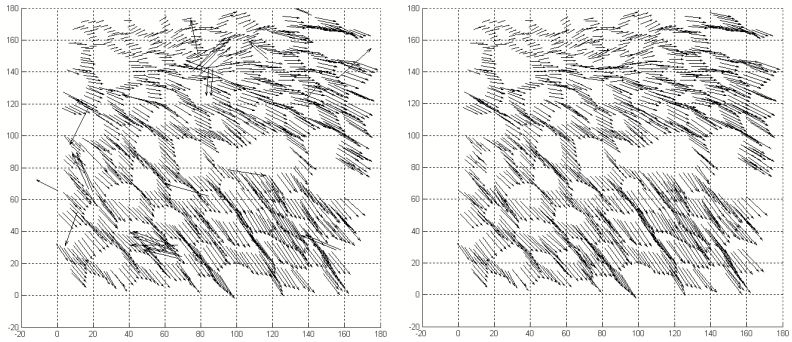


Figure 5.5: **Optical flow field**. *The two images refer to the optical flow vector field computed on 5.4. The raw computation (left) yields a wrong result. The smoothness constraint is not respected. The refined optical flow is instead a smooth vector field. See also images 5.4*

5.4 Case studies

5.4.1 Miocardial cells

We found the image sequences of miocardial cells to be noisy and poorly contrasted. The sampling frequency was low -a couple of frames per hour-, allowing the cells to undergo abrupt changes of shape. The illuminination changed from sequence to sequence, according to a number of parameters such as serum depth and purity, number of cells, chemical impurities and lightness condition of the lab.

Generalized Active Contours (see chapter 3) along with J-Maps showed to be a reliable tool. The workflow we used is to first segment the shape using active contours, then to evolve it over one time frame using J-Maps, and finally refine the result again by means of active contours. Using notation of equation 5.4, the initial Φ_0 is computed as discussed in section 3.3 while δ_t^{t+1} is obtained using J-maps followed by an active contours adaptation. The initial segmentation Φ_0 is achieved by means of Generalized

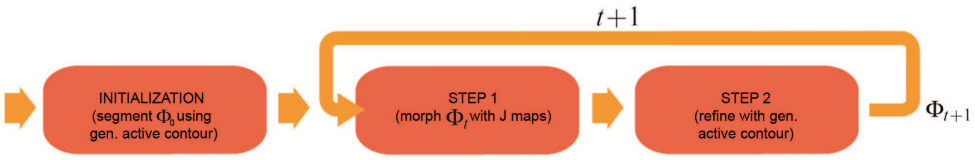


Figure 5.6: **Schematic workflow of dynamic single shape detection using J-maps and Generalized Active Contours**

Active Contours. For the each successive frame, rather than capture the shape from scratch again, we morph it according to equation 5.4. Here δ_t^{t+1} is provided by a two step approach: first computing J-maps and applying them to Φ_t, then refining the result using active Contours

Figure 5.7 shows results from a number of image sequences.

5.4.2 Drosophila epithelium

Each video sequences of the Drosophila epithelium contain 40-50 consecutives frames, obtained by means of a microscope. The patch contains 300-400 cells of different sizes. Cells split, merge together, deform and migrate; new cells enter the frame and old ones exit. To handle the plethora of behaviors, we extended the technique presented in the flowchar 5.6 of section 5.4.1. The resulting technique is a container of many of the techniques developed so far, such as random walk agent, Generalized Active Contours and J-maps. To initialize the algorithm, we detect the shape in frame 1 using the Random Walk approach as explained in section 4.4.1. The algorith provides a graphical representation of the shape. Using J-maps (as seen in section 5.2), we morph the graph according to equation 5.7, obtaining an approximation of the shape for the successive frame (see figure 5.9). We now convert the graph into an active Contour-like structure. To keep things simple and linear we omit the details of the conversion. In a nutshell, a reticular active Contour is similar to a curvilinear active

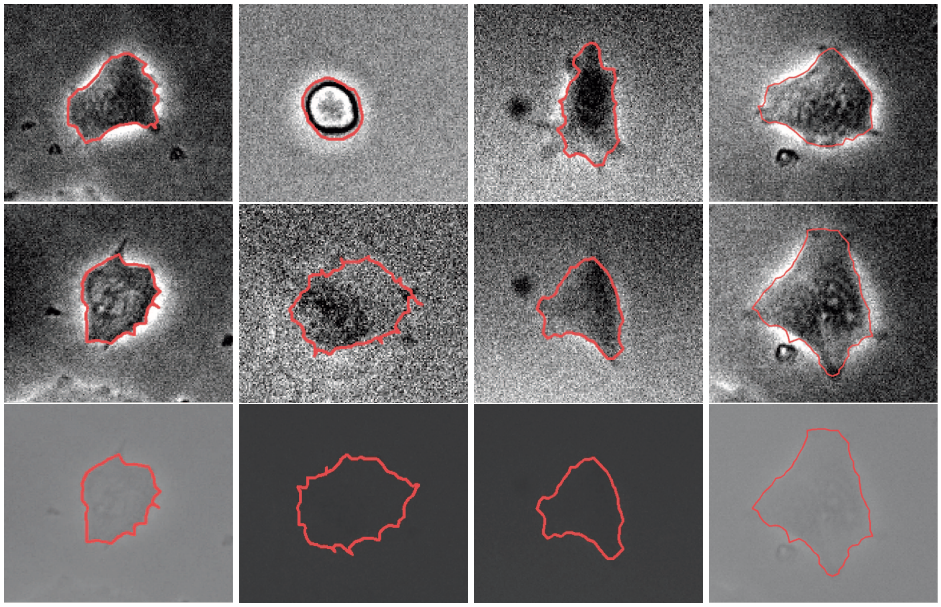


Figure 5.7: **Miocardial cell dynamic shape detection**. *We used the workflow in figure 5.6 to track cells. We employed the method on a total 10 sequences. The images show some frames taken from sequence number* 4, 5, 7 *and* 8, *respectively on column* 1,2,3 *and* 4. *Interestingly enough, each sequence has a different illumination and a different level of noise. The first two rows were manually contrasted by photoshop to the only purpose of clarity for this publication. The final row, instead, shows the raw video - the input of the algorithm*

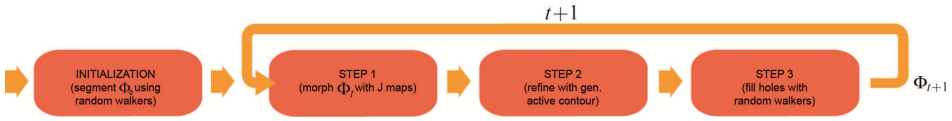


Figure 5.8: **Schematic workflow of dynamic single shape detection using J-maps and Generalized Active Contours**. *The initial segmentation* Φ_0 *is achieved by means of Generalized Active Contours. For the next frame, rather than capture the shape from scratches again, we morph it according to equation 5.4. Here* δ_t^{t+1} *is provided by a two step approach: first computing J-maps and applying them to* Φ_t, *then refining the result using active Contours*

Contour -the one described in section 3.2- and it is still driven by equations 3.4 and 3.5. Indeed, the integrals for the computation of the elastic energies has to take in account the reticular structure. To our knowledge, this is a new extention to the active Contours.

Evolving the Active Contour in a pseudo time removes false positives edges and refine the structure. Finally, we generate a set of random walk agent from each node

of the graph. This is necessary to fill holes and to "expand" the shape over the possibly appeared portion of the frame.

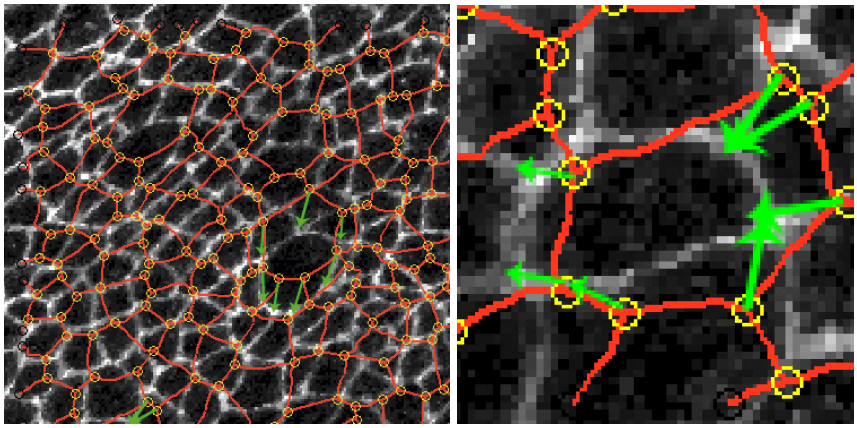


Figure 5.9: **Morphing of the Drosophila reticular structure using J-Maps**. *The image shows the reticular shape detected in frame* 11 *overimposed on frame* 12. *The green arrows schematically represents the morph obtained by means of J-maps, applied here to two cells*

Figure 5.10 shows results applied to a number of images of the drosophila wing epithelium.

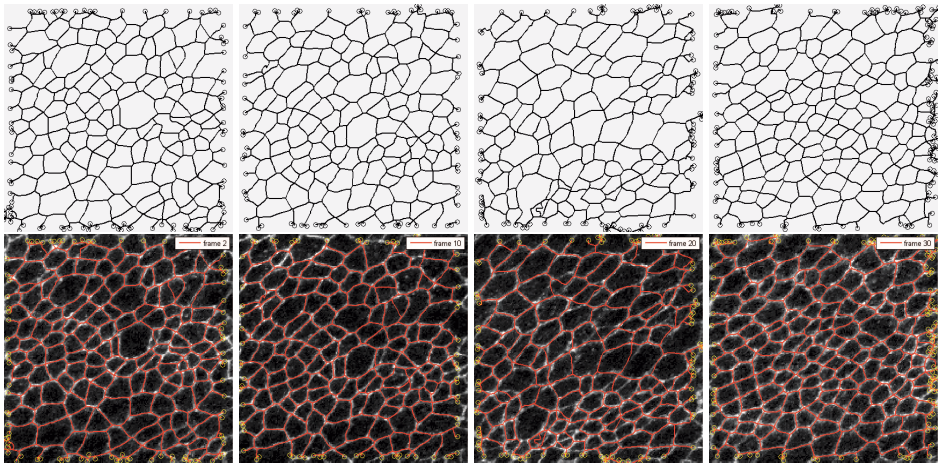


Figure 5.10: **Dynamic shape detection on the drosophila "wing" epithelium**. *The sequence is the same of image reffig:drosoRes, whereas the shape is here detected using the algorithm in figure 5.8, exploiting the temporal coherence. The top row shows the detected reticular shape for frame* 2,10,20 *and* 30. *The bottom row shows the network overimposed to the original frame*

6

Shape Analysis

The chapter deals with shape analysis. Section 6.1 explain what is *analysis*, while sections 6.2 and 6.3 discuss the analysis respectively of the miocardial cells and the drosophila epithelium. Section 6.4 analyzes the variability of human and algorithmic melanoma border identification. Section 6.5 closes the chapter analyzing the cardial patches.

6.1 What is shape analysis

Shape analysis is, generally speaking, the process of extracting metrics and descriptors from a single shape representation model or a sequence of them [40]. In a nutshell, given a set of shapes $\{\Phi_0,\ldots,\Phi_t,\ldots,\Phi_N\}$ and their representation $\{\mathcal{R}(\Phi_0),\ldots,\mathcal{R}(\Phi_t),\ldots,\mathcal{R}(\Phi_N)\}$, the purpose of the analysis step is to extract useful "numbers" and "figures". Generalizing equation 1.6, this means

$$\{\mathcal{R}(\Phi_0),\ldots,\mathcal{R}(\Phi_t),\ldots,\mathcal{R}(\Phi_N)\} \xrightarrow{analysis} \mathbb{R}^k \tag{6.1}$$

These values are forces, lengths, labels, descriptors, medical diagnosis, classifications etc [22]. Where the context is clear, we will use

$$\{\Phi_0,\ldots,\Phi_t,\ldots,\Phi_N\} \xrightarrow{analysis} \mathbb{R}^k \tag{6.2}$$

instead of 6.1. Equation 1.6 of section 1.2 simply considers $N = 1$. Roughly speaking, in fact, one can perform an analysis on the single shape, thus using information relative to the single time t, or on the whole set of shapes, exploiting the temporal information.

The measure of shape mainly serves two purposes: on the one side there is a *control purpose*, on the other there is a *classification purpose*, aiming at learning the shape representation.

Given a planar shape $A \subset \mathbb{R}^2$ a simple way to derive a measurement $f(A) \in \mathbb{R}^n$ is to refer to linear filters: $f(A) = \int \psi(A)da$, with $\psi(\bullet)$ suitable base functions [35].

A polynomial set of functions $(\psi(x,y) = x^p y^q)$ in the filter formulation produces the order $(p+q)$ curve moments $m_{p,q}(C)$. Although the moment description captures various levels of the shape distribution, single moments (especially the high order ones) link poorly to perceivable principal deformation, thus failing in supporting the perception with actual measurements.

$$f_i(S) = \int_S \Phi_i(x,y)dxdy \tag{6.3}$$

$$m_{p,q}(C) = \oint_C x^p(s) y^q(s)\,\delta(s)\,ds, \tag{6.4}$$

with δ linear density of the contour. Knowing the centre of mass of the curve, $x_C = \frac{m_{1,0}}{m_{0,0}}$ and $y_C = \frac{m_{0,1}}{m_{0,0}}$, allows making the moments invariant under **translation**, to produce the **central line moments** μ:

$$\mu_{p,q}(C) = \oint_C (x(s) - x_C)^p\,(y(s) - y_C)^q\,\delta(s)\,ds. \tag{6.5}$$

6.2 Miocardial cells analysis

Vitality of the cells is an important monitoring parameter to assess the quality of a colture. Miocardial cells, in particular, undergo a series of shape transformation over time: the magnitude of these changes is a direct measure of the vitality of the cells. One observes in fact that healthy cells evolve their shapes frequently, while cells in poor health exhibit quite a static behavior.

A computer aided system can detect a badly conditioned colture and react in response, for example adjusting temperature, chemicals and mechanical stresses. Figure 6.1 illustrates a feedback system in which a computer acquires images of the cells and proactively adjust the colture's environment [23].

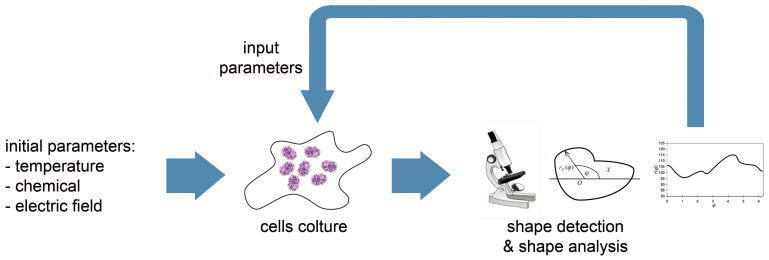


Figure 6.1: **Shape analysis and colture feedback**. *The quality of a cell colture depends on a number of parameters such as temperature, electric and magnetic fields, chemicals. A computer vision system performing shape detection and shape analysis can automatically regulate them to achieve a defined goal, such as maximizing the number of living cells, the overall vitality of the colture, or the growth rate*

In this context, then, we will try to translate the shape information into useful parameters conveying the "vitality" idea, Not fully satisfied by the meaningfullness

of the central moments, and to enforce consistency with the perception, we completed the set of shape descriptors introducing a measure of **sphericity** Ψ, **spikeness** and **boundary activity**. **Sphericity** is a scaled ratio between shape area $\mathcal{A}$ and perimeter Λ:

$$\Psi(\mathcal{C}) = \frac{2\pi \cdot \mathcal{A}(\mathcal{C})}{\Lambda(\mathcal{C})^2} \tag{6.6}$$

inspired by the definition of **ellipticity** and **ellipticity variance** given in [7].

We applied the classical definition of total **curvature** κ_{tot} [45] as

$$\kappa_{tot}(\mathcal{C}) = \oint_{\mathcal{C}} \left| \frac{\partial^2 \mathcal{C}(s)}{\partial s^2} \right| ds \tag{6.7}$$

which through normalization leads to the **Average Curvature**

$$\kappa_{avg}(\mathcal{C}) = \frac{\kappa_{tot}(\mathcal{C})}{\oint_{\mathcal{C}} \frac{\partial \mathcal{C}(s)}{\partial s} ds} \tag{6.8}$$

The quantities **spikeness** ξ and **boundary activity** υ are thus defined as

$$\xi(\mathcal{C}) = \oint_{\mathcal{C}} \left| \frac{\partial^2 \mathcal{C}(s)}{\partial s^2} - \kappa_{avg}(\mathcal{C}) \right| ds \qquad \upsilon(\mathcal{C}) = \left(\Lambda(\mathcal{C}) - \frac{1}{n} \cdot \sum_{j=t-n}^{t-1} \Lambda(\mathcal{C}_j) \right)^2 \tag{6.9}$$

where $\mathcal{C}_j$ refers to the shape $\mathcal{C}$ at time step j and n controls the temporal averaging. **Boundary Activity** makes explicit use of the time lapse for its computation.

The computation of these quantities benefits from the underlying proposed Snake model, in that it is more accurate and conveys more information about the dynamics of the shape and the deformation process. Table 6.2 shows differences in a number of metrics between the Generalized Active Contour and the classical one, on a test performed on the image sequence of image 6.2.

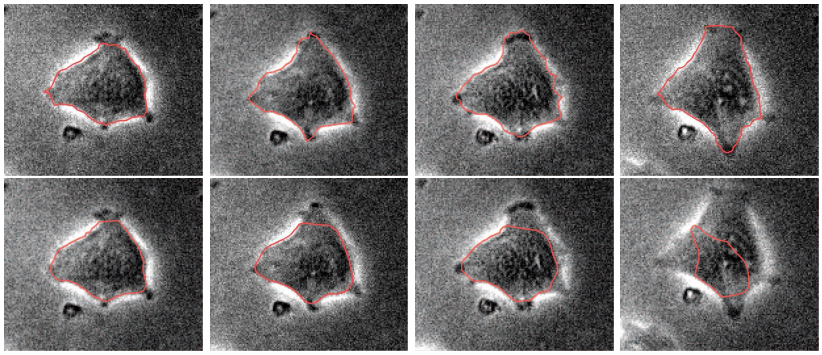


Figure 6.2: **Miocardial cell shape detection** *We repropose figure 3.5. The first row is obtained using the proposed Active Contour model, while the second row is produced with the original formulation*

Metrics	Generalized Active Contour			Classical Active Contour		
	frame 55	*frame 135*	*frame 185*	*frame 55*	*frame 135*	*frame 185*
Area	3092.0	3540.0	3852.0	2921.0	2984.0	2707.0
Perimeter	227.37	256.84	295.59	210.53	210.82	196.70
Area / Perimeter	13.60	13.78	13.03	13.87	14.15	13.76
Sphericity	0.751	0.674	0.554	0.828	0.843	0.879
Ellipticity	0.779	0.739	0.602	0.855	0.865	0.887
Ellipticity Variance	0.981	0.967	0.967	0.983	0.981	0.988
Total Curvature	483.1	485.7	877.8	334.1	258.0	220.16
Average Curvature	2.12	1.89	2.96	1.58	1.22	1.11
Spikeness	2.43	2.61	4.91	1.08	0.82	0.49
Boundary Activity	28.38	54.53	497.11	24.23	28.12	98.49

We consider in particular the parameter we have just introduced: **spikeness** and **boundary activity**.

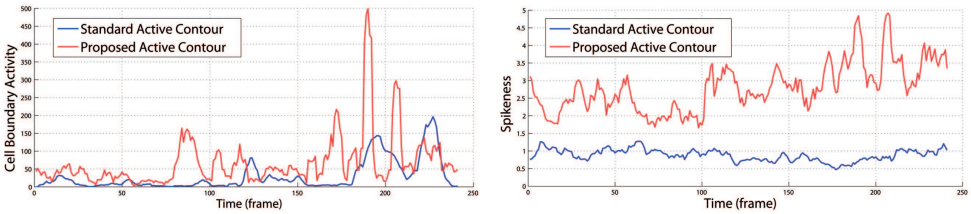


Figure 6.3: **Shape metrics.** *Evolution of spikeness and boundary activity, according to the proposed and original snake model*

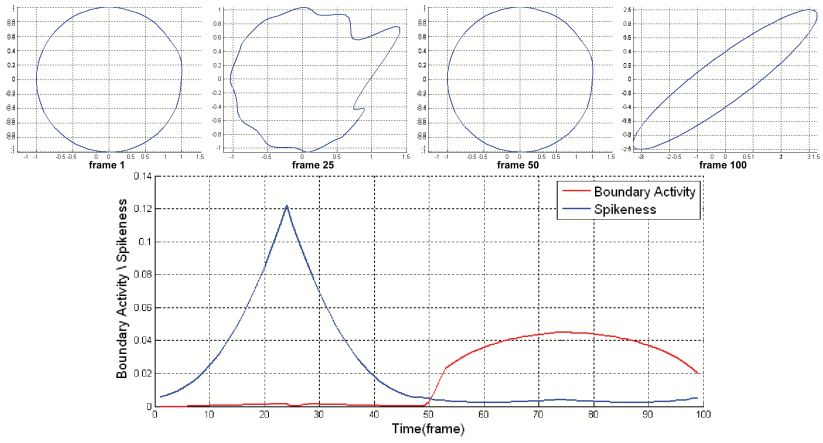


Figure 6.4: **Boundary Activity and Spikeness synthetic shape test**. *The figure shows the **boundary activity** (red) and the **spikeness** (blue) measures computed on a 100-frames-long synthetic video sequence. We build the test to highlight the differences between the two and show how they can be decoupled*

To better understand the means of these parameters, and the validity of the model that allows their computation, we run a number of synthetic test. In the first test (figure

6.4), a circular shape (frame 1) deformes into a spiky shape (frame 25) and then back again to the initial one (frame 50). As expected, the **spikeness** line has a maximum on frame 25. On the other side, being the length of the perimeter almost unaffected by the local transformation, the **boundary activity** keeps close to zero. From frame 50, the circle becomes an ellipse. The **spikeness** measure now stays close to zero, while the **boundary activity** shows a significant rise.

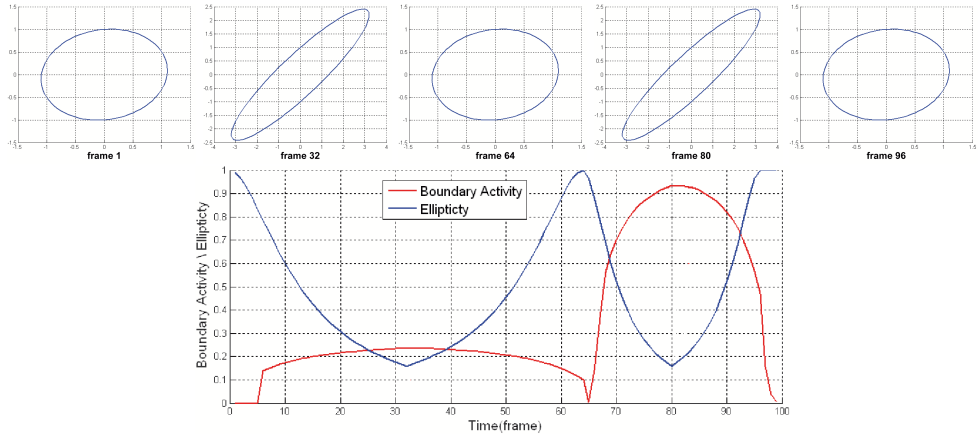


Figure 6.5: **Boundary Activity and Ellipticity synthetic shape test**

In the second test, on figure 6.5, a circle (frame 1) morphs into an ellipse (frame 32) and then back again into a circle (frame 64). The same transformation applies again from frame 64 to 96, at a double speed. The **ellipticity** measure (blue) scores from 0.97 to 0.18, with minima and maxima points of the same value. On the contrary, the **boundary activity** (red) reaches an higher value during the second faster transformation.

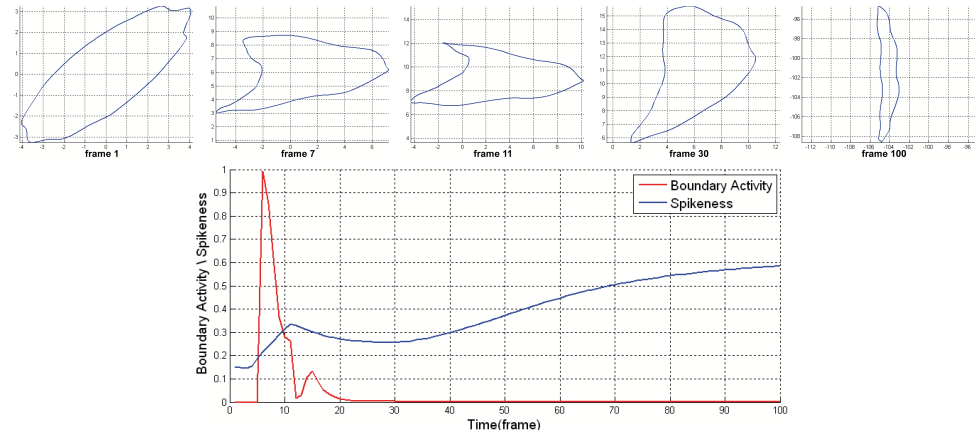


Figure 6.6: **Boundary Activity and Spikeness synthetic shape test (2)**

Figure 6.6 shows our final test, a complex transformation. The shape undergoes

abrupt changes from frame 1 to 11 (detected by the high values of the red **boundary activity** line), and then a constanst and slow stretch (frames 30 and 70. All these tests generate shape synthetically and morph them in time according to a known transformation, whose intensity and direction can be controlled. The analysis reflects these transformations. The goal is to develop a small set of measures able to decouple the many effects of a transformation. In a nutshell, even if a shape deforms in a complex way (such as in figure 6.6), measures recover from the complexity and give a simplified yet consistent analysis.

6.3 Drosophila ephitelium

We peformed a twofold analysis. On a single frame, we extracted the network structure removing irrelevant information. On a sequence level, we correlated the sets of obtained network structures using a point set matching algorithm.

6.3.1 Network extraction

The information gathered by the Random Walk Agents may be overabundant for modeling purposes, especially when the goal is to obtain compact-size models, flexible and agile, to be used both for simulation and for prediction of the behavior of the structure. Each location reached by a random walk agent is a node in the graph. Having a small parameter k in equation 4.1 thus leads to a very precise but dense graph. In this respect, only the subset $\mathcal{N}_r$ of the nodes associated to non-scalar Θ (and the correspondent edges $\mathcal{E}_r$) are interesting in the description, that is:

$$\mathcal{G}_r = (\mathcal{N}_r, \mathcal{E}_r)\ \mathcal{N}_r \subset \mathcal{N}\, s.t. \dim(\Theta(n_i)) > 1 \forall n_1 \in \mathcal{N}_r,$$

where $\dim(\Theta(n_i))$ indicates the cardinality of vector Θ for node n_i. If the border effects are neglected, the resulting graph $\mathcal{G}_r$ is a 3-connected graph, in which the minimum vertex degree (valency of the graph vertices) is $d = 3$.

In a nuthsell, reconstruction of the geometric model of the cells is carried on considering only nodes with degree greather than 2. These nodes are considered the **corner** of the edges of the cells. The new graph $\mathcal{G}_r$ consists of such corners and new edges created between a pair of corners if in the original graph there was a *straight* path (a path connecting them not passing through any other corner). By doing so, we prune a lot of nodes and edges and end up with a light, compact representation of the cellular structure.

Figure 6.7 shows results of the network simplification procedure applied to (a small portion of) frames 13 and 42.

6.3.2 Point set matching

The general problem of point set matching is a fundamental topic in computer vision and is key for the registration of multiple images. We developed a general, non-

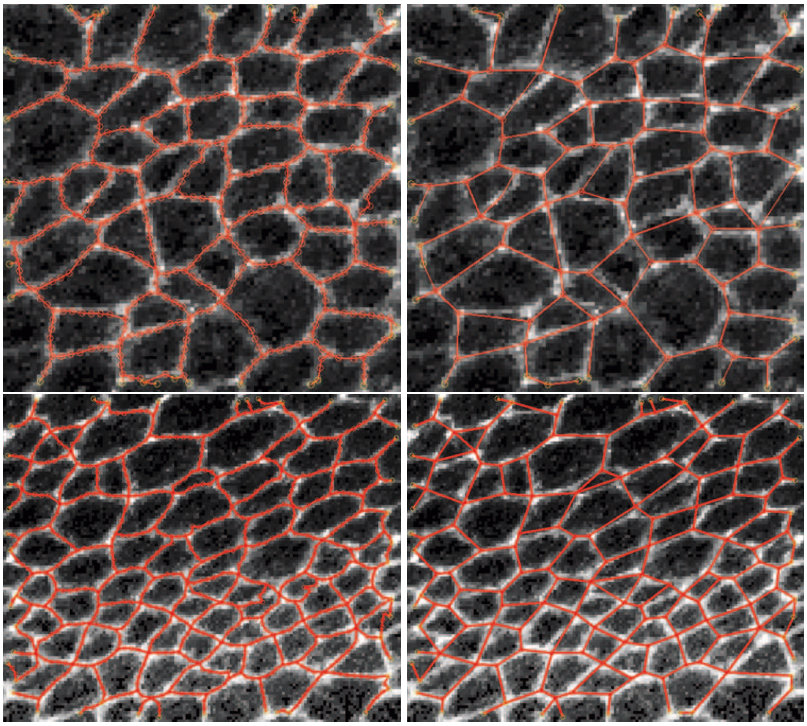


Figure 6.7: **Network simplification**. *Images show as circle the vertices of the graph. The first row, computed on frame* 13 *of the drosophila "wing" video sequence, shows a very dense structure. Each node corresponds to a point traced by a random walk agent. The structures on the right is the simplified graph, where only nodes with degree superior than* 3 *were kept. This leads to a "straightening" of the edges and to a more compact representation. The bottom row shows the same simplification on frame* 42

iterative technique based on spectral methods. The purpose of this section is to illustrate the approach, and then apply it to the drosophila image sequences. We make use of a *pairing matrix*, which relates pairs of points taken from the two sets. In addition, we allow the elements of the pairing matrix to depend on a combination of possible metrics, each of which is defined over the two sets of points.

The technique performs robustly on a variety of application domains, by automatically adapting the set of useful metrics to the particular case under study. Specifically, the multiplicative structure of the pairing matrix ensures that if a particular metric is not discriminative enough for a pair of images (versus other similarity measures), it is automatically overridden by the other more relevant metrics.

Consider two sets of points $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, with elements lying in $\mathbb{R}^d$ for some finite d, and not necessarily with the same cardinality. We introduce a *pairing matrix* $Z \in \mathbb{R}^{mn}$, with entries $z_{ij} \in \mathbb{R}$. Each element z_{ij} is intended to express a measure of similarity between point $x_i \in X$ and $y_j \in Y$ and will be specified

shortly.

Given a matrix $\mathcal{Z}$, the selection of pairs of matching points from the two sets is performed after a normalization procedure: the matrix $\mathcal{Z}$ is preprocessed by computing its singular-value decomposition

$$\mathcal{Z} = TDU, \tag{6.10}$$

where $D \in \mathbb{R}^{mn}$, whereas T,U are properly-sized orthogonal matrices. We replace diagonal elements d_{pp} of $D, p = 1,2,\ldots,\min\{m,n\}$, with identity constants, which yields

$$\widetilde{\mathcal{Z}} = TEU, \tag{6.11}$$

where $e_{ij} = d_{ij}, i \neq j, i = 1,\ldots,m, j = 1,\ldots,n, e_{pp} = 1, p = 1,\ldots,\min\{m,n\}$. This technique is generally known as *whitening*. The largest elements in $\widetilde{\mathcal{Z}}$ correspond to candidate matching pairs as follows: the pair (x_i,y_j) is matched if and only if z_{ij} is the largest element both of row i and of column j. This strong correspondence implies a "mutual consent" to the match: indeed, if z_{ij} is the largest element of row i but not of column j, point x_i is similar to y_j, but not the contrary. As such, the pair (x_i,y_j) is not a valid match.

If one considers each row i as an n dimensional vector r_i, then $\mathcal{Z}$ is a map from point x_i into vector r_i. Ideally, a pairing matrix $\mathcal{Z}$ should be sparse, with a single non-zero element per row and linearly independent rows. In such a case, r_i coincides with one of the coordinate axis of $\mathbb{R}^n$ and as such is the farthest possible from any other vector row. More generally, if a row vector r_i is close to a coordinate axis $e_j \in \mathbb{R}^n$, then it is likely for the pair (x_i,y_j) to be a match. However, if two vectors r_{i_1} and r_{i_2}, $i_1,i_2 \in \{1,\ldots,m\}$, are adjacent to the same coordinate axis e_j, then the corresponding points x_{i_1} and x_{i_2} "compete" for the match with y_j. Setting the singular values of $\mathcal{Z}$ to be unitary corresponds to a spatial outspread of its row vectors, thus alleviating such potential conflicts. Figure 6.8 displays an instance of $\mathcal{Z}$ and $\widetilde{\mathcal{Z}}$ matrices corresponding to sets of cardinality three. In this example where $m = n$, the above operation corresponds to the normalization the volume of the associated prism.

Let us consider the following set of metrics $d_k : X \times Y \to \mathbb{R}, k = 1,\ldots,5$:

Metric	**Definiton**
$d_1(x_i,y_j)$	$\|x_i - y_j\|_p, p > 0$
$d_2(x_i,y_j)$	$\cos(\mathtt{m}(i),\mathtt{m}(j))$
$d_3(x_i,y_j)$	$\cos(\tilde{\mathtt{m}}(i),\tilde{\mathtt{m}}(j))$
$d_4(x_i,y_j)$	$\lvert\mathtt{d}(x_i) - \mathtt{d}(y_j)\rvert$
$d_5(x_i,y_j)$	$\mathtt{td}(x_i,y_j)$

1. d_1 is a p-norm distance between pairs of points taken from the two sets. The present case studies consider the Euclidean norm ($p = 2$).

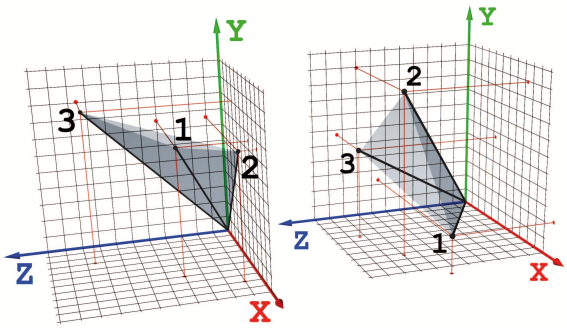


Figure 6.8: **A withening example**. *Graphical comparison of pairing matrix* z *(left) and its normalized version* $\widetilde{z}$ *(right), where* $z = \begin{bmatrix} 0.7000 & 0.7500 & 0.4000 \\ 0.9000 & 0.9500 & 0.1000 \\ 0.3000 & 0.9000 & 0.8500 \end{bmatrix}$ *and* $\widetilde{z} = \begin{bmatrix} 0.8519 & 0.1527 & 0.5010 \\ 0.4249 & 0.7607 & 0.4907 \\ 0.3062 & 0.6309 & 0.7129 \end{bmatrix}$. *Notice that the row vectors* r_1, r_2, r_3 *of* $\widetilde{z}$ *do not cluster and better spread in space. The pairing matrix* z *yields one pair* $(2,2)$*, whereas* $\widetilde{z}$ *yields the pairs* $(1,1),(2,2),(3,3)$

2. d_2 is a measure of the distance of the i-th *mode* $\mathtt{m}$(i) and the j-th *mode* $\mathtt{m}$(j) associated to each of the two sets. As suggested in [49], the mode $\mathtt{m}$ of a point set is computed as follows: first, a square proximity matrix defined according to the intra-distances between the features of the image is introduced; it is successively diagonalized and its first $\min\{m,n\}$ eigenvectors, sorted according to the largest eigenvalues, are regarded as its modes. The distance between the modes $\mathtt{m}$(i) and $\mathtt{m}$(j) of the two graphs is then computed as their cosine.

3. This metric is valid if the sets X,Y are embedded with a graphical structure. The distance d_3 is characterized analogously as d_2, however the modes $\tilde{\mathtt{m}}$ are computed by considering a proximity matrix that has non-zero elements only if the corresponding pair of vertices are connected by an existing edge in the graph.

4. The metric d_4 is defined as the absolute value of the difference between the graphical degree of a pair of points, taken respectively from X and from Y. The function $\mathtt{d}$ is the degree of a node in a graph. As for d_3, this metric is valid if the sets X,Y are embedded with a graphical structure. However, if no graphical structure is present over the sets X and Y, then one such graph can be induced artificially: for instance, edges can be created between pairs of points if their distance is less than a tunable threshold.

5. Provided an image underlying the point sets, the metric d_5, defined by a function $\mathtt{td}$, computes the *texture difference* between a neighborhood of $x_i \in X$ and one of $y_j \in Y$. Here the points x_i, y_j are intended as features of the corresponding

images.
More formally, let us consider the finite discrete domain $L \subset \mathbb{Z}^2$ made up of the pixels of the two-dimensional image frame. Given a point $z \in \mathbb{R}^2$, we define a neighborhood $\mathcal{N}(z,\delta) \subset \mathbb{Z}^2$ as the set of pixels of the image with centroid lying within a radius $\delta > 0$ from z. A function $I : L \to \mathbb{R}^+$ specifies the *intensity* of the image over its domain.
Let us now consider the two images L_x, L_y underlying X, Y. Given a radius δ of choice, the function $\mathtt{td}(z_1, z_2)$ computes the absolute value of the difference between the intensities of the pixels in the neighborhoods of points z_1 and z_2:

$$\mathtt{td}(z_1,z_2) = \sum_{p_1 \in \mathcal{N}(z_1,\delta) \cap L_x} \; \sum_{p_2 \in \mathcal{N}(z_2,\delta) \cap L_y} |I(p_1) - I(p_2)|.$$

This approach is related to a similar procedure used in [51].

Consider the two sets of feature points X and Y, with cardinality m and n respectively. The outcomes of the point set matching procedure is compared with the ground truth. The ground truth is known for the synthetic case studies, whereas for the case studies based on real images it is directly assessed over the data sets by independent and unbiased observers. Let us introduce the following four entities:

1. $X_{tm} \subseteq X, Y_{tm} \subseteq Y$ are the two sets of feature points that are correctly matched, of cardinality respectively m_{tm}, n_{tm}
2. $X_{ts} \subseteq X, Y_{ts} \subseteq Y$ are the two sets of feature points that are correctly left unmatched, of cardinality respectively m_{ts}, n_{ts}
3. $X_{fm} \subseteq X, Y_{fm} \subseteq Y$ are the two sets of feature points that are wrongly matched, of cardinality respectively m_{fm}, n_{fm}
4. $X_{fs} \subseteq X, Y_{fs} \subseteq Y$ are the two sets of feature points that are wrongly left unmatched, of cardinality respectively m_{fs}, n_{fs}

Notice, as intuitive, that $m_{tm} + m_{ts} + m_{fm} + m_{fs} = m$ and that $n_{tm} + n_{ts} + n_{fm} + n_{fs} = n$. We define as percentages, over both sets of points, the following quantities:

1. *true matches*, as the ratio of feature points that are correctly matched, i.e. $\frac{m_{tm}+n_{tm}}{m+n}$
2. *true singles*, as the ratio of feature points that are correctly left un-matched, i.e. $\frac{m_{ts}+n_{ts}}{m+n}$
3. *false matches*, as the ratio of feature points that are wrongly matched, i.e. $\frac{m_{fm}+n_{fm}}{m+n}$
4. *false singles*, as the ratio of feature points that are wrongly left un-matched, i.e. $\frac{m_{fs}+n_{fs}}{m+n}$

The outcome of the case studies will be evaluated according to the introduced quality measures.

Synthetic Graph Matching

We consider a graphical structure $G = (V, E)$ with two dimensional spatial components, which are constrained to lay within the unit square in the positive quadrant $[0,1]^2$. The cardinality of the set of nodes V is equal to 50 and their spatial components are generated uniformly at random within the specified domain. The edge set E is created between pairs of nodes in V according to a Bernoulli distribution with mean equal to 0.5, however the edges that are longer than a specified threshold (0.25) are discarded. Figure 6.9 shows an example of a graph.

The graph is then morphed into a new structure $\tilde{G} = (\tilde{V}, \tilde{E})$, according to the following procedure:

1. The set $\tilde{E} \subseteq E$ is generated from E by discarding each edge according to a Bernoulli probability distribution with mean $\mathbb{P}_e$;

2. The set $\tilde{V} \subseteq V$ is generated from V by discarding each edge according to a Bernoulli probability distribution with mean $\mathbb{P}_v$, and additionally by eliminating the residual edges that connect to vertices in $E \setminus \tilde{E}$;

3. The spatial components associated to the elements in $\tilde{E}$ are obtained from those belonging to the corresponding elements in E by perturbing them with the addition of a random variable that is uniformly distributed within the square $\frac{\mathbb{M}_v}{2}[-1,1]$. In other words, the original coordinates are subjected to a uniform perturbation that amounts to $\mathbb{M}_v$% of their maximum possible value.

The matching procedure proposed is tested on a cohort of pairs of graphs $(G, \tilde{G})$, parameterized by the input configuration $(\mathbb{P}_e, \mathbb{P}_v, \mathbb{M}_v)$ used to generate them: for each combination $(\mathbb{P}_e, \mathbb{P}_v, \mathbb{M}_v)$ of perturbation parameters we average the outcomes of 2000 simulations.

Figures 6.9 and 6.10 represent respectively a single pair of test graphs ($\mathbb{P}_e = \mathbb{P}_v = \mathbb{M}_v = 15\%$), and the outcomes of the matching procedure.

We have employed the first four metrics d_1, d_2, d_3 and d_4. The metric d_5 is not employed, since the artificial graphs have no underlying physical image that can be exploited for the matching procedure. Table 6.1 reports the results for each configuration of the perturbation parameters $(\mathbb{P}_e, \mathbb{P}_v, \mathbb{M}_v)$ are the average of the 2000 simulations. The experiments are divided in five batches: in the first, we uniformly modify the three perturbation parameters; in the following four, we fix two of the three parameters and modify the remaining one by using values that match those of the first batch of experiments. The monotonicity of the performance outputs of the algorithm with respect to the perturbation level provides evidence of the consistency of the procedure (see first set of simulations). The results of the last four groups of simulations "lie within" those of the first batch (the comparison ought to be done by looking at the accrued true and false value pairs). By comparing the third result of each group of experiment with the first two, one can observe that the elimination of edges or vertices affects the quality of the outcomes more than the perturbation of the spatial coordinates of the vertices. This is despite the fact that spatial perturbations can result in

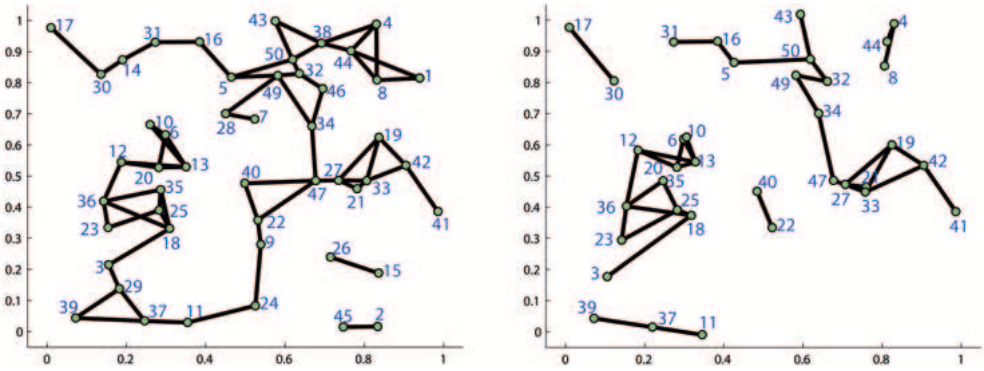


Figure 6.9: **Synthetic graph generation and perturbation**. *The original graph (left) contains* 50 *nodes. The perturbed graph (right) was generated using* $\mathbb{P}_e = \mathbb{P}_v = \mathbb{M}_v =$ 15%. *The blue labels mark the nodes and provide the ground truth correspondence*

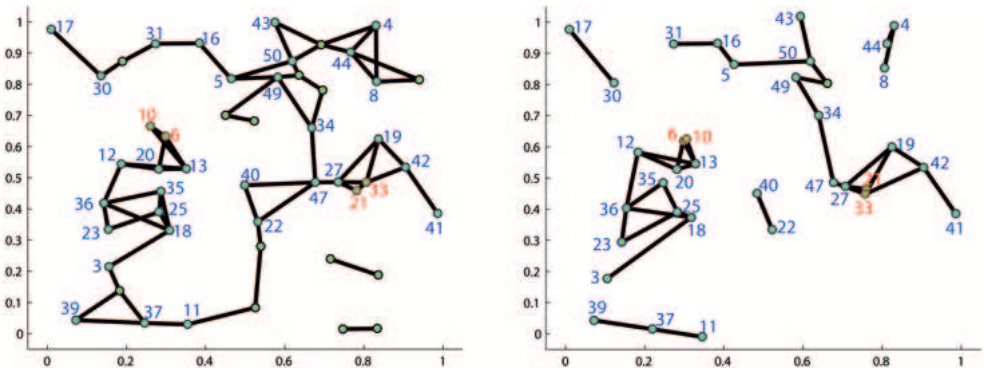


Figure 6.10: **Outcomes of the matching procedure**. *Graphs refer to the synthetic ones of Figure 6.9. Here the blue labels indicate correctly matched nodes (true matches), whereas red labels shows the wrongly matched ones. Nodes without labels are correct single nodes*

feature crossover. Furthermore, as intuitive, the elimination of an edge (first result of the last four groups) affects the results more than that of a vertex (second result).

Point Set Matching over a Literature Benchmark: The "CMU House"

We have finally tested our procedure on a known benchmark from the computer vision literature, known as the "CMU House" [1]. This benchmark contains a set of 110 pictures of a toy house, taken over a black background. We have extracted a set of features from each image by applying a corner detector [56]. The obtained sets have a cardinality that is very similar to the sets used in [10, 54] for the same benchmark, which allows for a fair comparison with those results.

We have tested our algorithm on two experimental setups. Firstly, we have matched

Perturbation			**Output Performance**			
$\mathbb{P}_e$	$\mathbb{P}_v$	$\mathbb{M}_v$	true matches	true singles	false matches	false singles
15.00%	15.00%	15.00%	71.01%	7.50%	19.37%	3.12%
12.00%	12.00%	12.00%	79.83%	6.66%	12.07%	1.44%
9.00%	9.00%	9.00%	87.65%	4.37%	7.10%	0.88%
6.00%	6.00%	6.00%	93.06%	3.38%	3.36%	0.20%
3.00%	3.00%	3.00%	97.09%	1.82%	1.04%	0.05%
12.00%	15.00%	15.00%	72.40%	7.21%	18.59%	2.80%
15.00%	12.00%	15.00%	73.17%	6.05%	18.64%	2.14%
15.00%	15.00%	12.00%	78.07%	8.94%	12.19%	1.90%
9.00%	12.00%	12.00%	80.29%	6.71%	11.41%	1.59%
12.00%	9.00%	12.00%	82.31%	4.32%	12.05%	1.32%
12.00%	12.00%	9.00%	84.49%	6.64%	7.80%	1.07%
6.00%	9.00%	9.00%	87.86%	5.62%	6.03%	0.49%
9.00%	6.00%	9.00%	89.31%	3.32%	6.72%	0.65%
9.00%	9.00%	6.00%	90.86%	5.23%	3.42%	0.59%
3.00%	6.00%	6.00%	93.26%	3.60%	2.78%	0.36%
6.00%	3.00%	6.00%	95.17%	1.68%	2.99%	0.16%
6.00%	6.00%	3.00%	94.86%	3.70%	1.20%	0.24%

Table 6.1: Outcomes of the matching procedure tested on sets of randomly generated and successively perturbed graphs. We have run 2000 simulation for each configuration of perturbation parameters and reported the average of the outcomes. For each of the 2000 simulations we have first generated a graph, then perturbed it. The perturbation level is tuned via three parameters: $\mathbb{P}_e$, the probability that an edge is erased from the original graph; $\mathbb{P}_v$, the probability that a vertex is eliminated from the original graph; $\mathbb{M}_v$, the level of spatial perturbation applied to a vertex of the original graph.

the 109 sequential pairs of images (1-2, 2-3, ..., 109-110). Figure 6.11 displays the output of one such pairing: the green labels are obtained from the matching procedure. Secondly, we have matched 109 pairs of distinct images, randomly chosen from the set. Figure 6.12 displays the output of one such matching: in green are the correct labels obtained from the matching procedure, whereas in red are the wrong outcomes.

The first study is meant to test the robustness of the method with respect to positional jitter, while the second targets the performance against large transformations and the presence of feature occlusions. Table 6.2 displays the results as averages over the 109 tests. The outcomes of both studies appear to sensibly improve those in [10], and to remarkably improve those in [54]. Notice that the performance measure in [10, 54] is based exclusively on the second component of the pair of images, and hence slightly differs from the one used in this work, which we believe is more accurate. Also, the statistics in both [10, 54] are quite limited in sample size and image range.

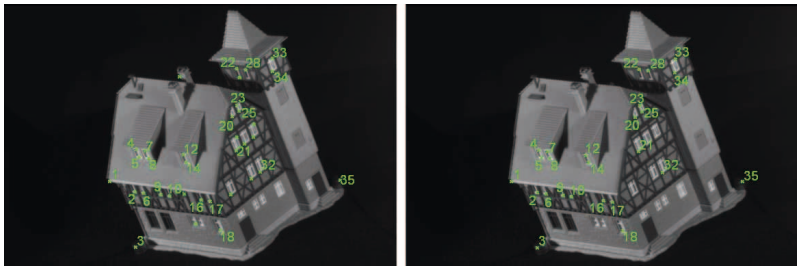


Figure 6.11: **CMU house point set matching test**. *The figure shows two successive images (frames 1 and 2) from the CMU House benchmark [1]. The green labels are obtained from the matching procedure. The outcome is, in this case, perfect*

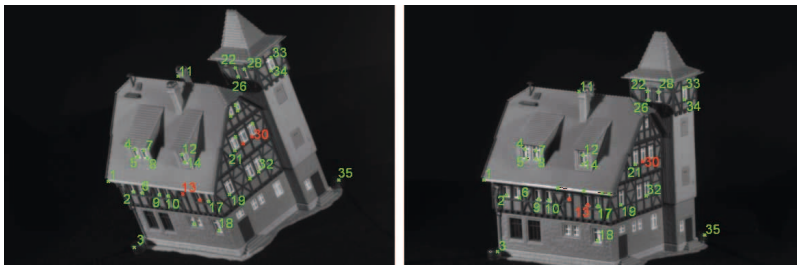


Figure 6.12: **CMU house point set matching test**. *Two random images (frames 1 and 67) are considered for the matching procedure over the CMU House benchmark [1].As for figure 6.11, the green labels correspond to correctly matched points (true matches and true singles), whereas the red labels mark wrong outcomes (false matches and false singles)*

Input	**Output Performance**			
image pairs	true matches	true singles	false matches	false singles
sequential	93.32%	5.48%	0.73%	0.47%
random	75.86%	16.20%	5.68%	2.26%

Table 6.2: **Outcomes of the matching procedure.** *Outcome of the matching procedure tested on 109 pairs of feature sets extracted from 110 images in [1]. The results are averages over the 109 tests. The top line refers to the 109 pairs of sequential images such as in figure 6.11, whereas the bottom one to 109 pairs of randomly extracted images, such as figure 6.12*

The Drosophila Wing

This experimental study aims at matching two network structures extracted from biological data.[1] Each network describes the cellular epithelium of a wing of *Drosophila*

[1]The images have been provided by the Axelrod Lab, at the Department of Pathology, Stanford University School of Medicine, Stanford, USA. Members of the Lab have also contributed in the inter-

melanogaster, the common fruit fly. The experimental data are obtained with confocal microscopy techniques a few hours after puparium formation. It is of interest for the developmental biologist to have access to quantitative data relating to the network structure of the epithelium. The graphical structure is extracted from single frames that belong to time-lapse movies of the epithelium. The details of the computer vision technique used to extract the network from a single frame are formally presented in [50]. Along with the collection of the graphical structures corresponding to each frame, it is important to match the networks extracted from pairs of frames that are successive in time. This procedure is also known as the *registration* of the frames of the movie.

The experimental data consist of 50 frames consecutive in time, corresponding to 49 pairs of images. Figure 6.13 shows frames (frame 22 and 23) taken from the wet lab experimental data.

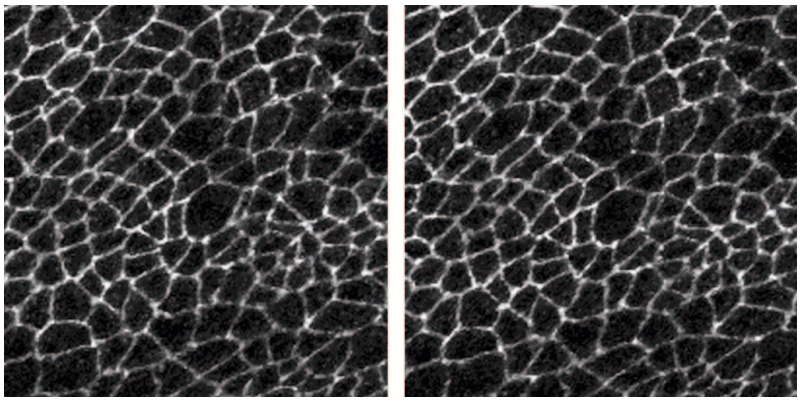


Figure 6.13: **Frames** 22 **and** 23 **considered for the matching procedure**. *The images are part of a 40 frame movie and refer to a section of the epithelium of the Drosophila melanogaster wing. The polygonal structures are 2-d sections of the epithelial cells*

For the instance under study, we have employed the metrics d_1 and d_5. The use of d_5 is dictated by the availability of an actual image containing meaningful information for the matching. The *intra-metrics* d_2, d_3 and d_4 are discarded, which is explained by observing the similarity of the neighborhood structure for most of the nodes in the graph. In other words, if most of the internal nodes have a similar number of connected edges, then the information provided by the metric d_4 is redundant. Similar considerations hold for the metrics d_2, d_3.

Figure 6.14 and 6.15 display the graphical structures extracted from the frames in Figure 6.13, and labeled with the outcome of the matching procedure. Unlike the study in Section 6.3.2, the ground truth has been provided by manual observation from an independent and unbiased observer. Table 6.3 displays the outcomes of the procedure, averaged over the 49 tests. The results appear to be rather good when compared to others in the biology literature [37], especially given the complexity of

pretation of the outcomes of the registration procedure.

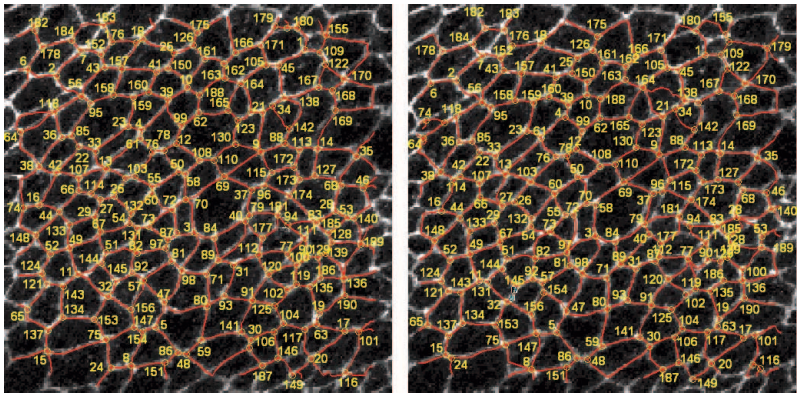


Figure 6.14: **The matching procedure applied to the networks of Figure 6.13**. *The yellow labels over the nodes correspond to matches (both correct and wrong ones). The unlabelled nodes are single nodes (both correct and wrong ones)*

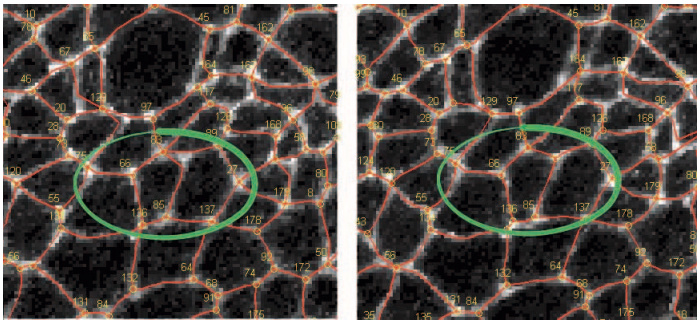


Figure 6.15: **A particular of the matching procedure from Figure 6.14**. *The structure undergoes a significant topological change. The green circle highlights a difficult match that is correctly resolved*

the structures and of the dynamics under study (cells both appear do to divide and to exit the epithelium, the frames are subject to translation, and the images are quite noisy).

Output Performance			
true matches	true singles	false matches	false singles
88.23%	3.61%	7.92%	0.24%

Table 6.3: **Outcomes of the matching procedure for the Drosophila test case**. *The matching procedure was here tested on 49 pairs of networks extracted from 50 successive frames of a movie. The results are averages over the 49 tests. The movie refers to the morphogenesis and the dynamics of a section of the wing of Drosophila melanogaster*

6.4 Variability in melanomas border identification

We analyze the performances of the color segmentation algorithm decribed in section 3.3.2 applied to images of melanocytic lesions. In particular, we compare the output provided by the algorithm to the output provided by dermatologists, in a Turing-test like fashion. In this context, *segmentation* means the classification of all points in the images as part of the lesion or part of the surrounding, healthy skin.

Unfortunately, segmentation of melanocytic lesions is a surprisingly difficult task. The fundamental reason lies in the fact that lesion borders are often fuzzy and there exists no operative definition of whether a portion of skin belongs to a lesion or not. Dermatologists rely on subjective judgement. Automated systems attempt to replicate the assessment of human dermatologists through a number of heuristics. Not surprisingly, this leads to appreciable variability in the localization of the precise border of lesions, not only between automated systems and human dermatologists, but also between different human dermatologists [27].

Quantifying this variability is crucial for at least two reasons. First, it allows one to estimate the level of noise affecting large, multi-operator epidemiological studies e.g. correlating lesion size to benignity. Second, human inter-operator variability effectively provides an upper bound to the segmentation accuracy achievable by any automated system, as long as "ground truth" is provided by the subjective evaluation of human dermatologists rather than by a standard operative definition. For example, if even experienced dermatologists disagree on how to classify 5% of the area of an image, no automated system can be expected to classify "correctly" more than 95% of the area of that image.

This section thus evaluates the variability in lesion border identification by a group of 12 dermatologists and 4 algorithms. Our is the largest of the studies so far, and the only one that differentiates dermatologists based on dermatoscopy training experience. The human inter-operator variability is then compared to the segmentation accuracy of the algorithms, representative of the three fundamental state-of-the-art automated segmentation techniques and of a fourth, novel, technique.

While some studies (e.g. [47]) have one or more dermatologists subjectively assess the quality of the proposed automated systems, the general consensus is that evaluation methods striving for a greater degree of objectivity are preferable [14]. Most of these methods rely on a *ground truth* segmentation against which the proposed segmentation is assessed, labeling its pixels as True Positive (TP), False Positive (FP), False Negative (FN) or True Negative (TN), depending on whether they are classified as part of the lesion, respectively, in both segmentations, only in the proposed segmentation, only in the ground truth segmentation, or in neither of the two. The number of pixels in the FP and FN categories, usually normalized dividing them either by the size of the proposed lesion (TP+FP), by the size of the ground truth lesion (TP+FN) or by the size of its complement (FP+TN), provide a measure of the divergence between the proposed segmentation and the ground truth.

The fundamental problem with these approaches is that any definition of "ground

Table 6.4: Some common metrics to evaluate segmentation

XOR Error [12]	$(\frac{FP+FN}{TP+FN}) \times 100\%$
Specificity	$(1 - \frac{FP}{FP+TN}) \times 100\%$
Sensitivity or Recall	$(1 - \frac{FN}{TP+FN}) \times 100\%$
Precision	$(1 - \frac{FP}{FP+TP}) \times 100\%$

truth" based on the segmentation of a single dermatologist is inherently highly subjective. Thus, several recent approaches combine the evaluation of multiple dermatologists to obtain a more objective ground truth segmentation. However, these approaches are more complex, and all exhibit some shortcomings.

All of mentioned metrics share a subtler but serious problem: they do not provide an idea of how well one can expect a proposed segmentation to perform. While some of them are normalized in such a way that, for every (set of) ground truth(s), the best score a segmentation can achieve is exactly 1, it is generally unrealistic for any human dermatologist - and thus for any automated segmentation system - to achieve such a score.

The solution we propose is simple: when evaluating an automated segmentation system, in addition to the ground truth segmentation(s), one should always employ one more "calibration" segmentation provided by an experienced dermatologist. The divergence of the calibration segmentation from the ground truth (by whatever metric one may choose) provides a clear, intuitive indication of the best divergence one can hope for when evaluating by that same metric an automated segmentation system (or even, in fact, a less experienced dermatologist!). *In a nutshell, we propose the variability between experienced human dermatologists in the localization of melanocytic lesion border to be used as a* gold standard *to assess the quality of any automated segmentation system.*

While the choice of the basic divergence metric is relatively unimportant, our choice would fall on the average of $\frac{FP}{FP+TP}$ over all ground truths (i.e. the Misclassification probability of [27]) paired with the complementary average of $\frac{FN}{FP+TP}$ to account for false negatives. The latter metric is similar to Precision and Recall, but the normalization takes place over the size of the lesion according to the segmentation under test (as in [27]) rather than according to ground truth – allowing divergence from each ground truth to have the same weight. This pair of metrics makes extremely clear the source of a segmentation's divergence from ground truth - identifying whether the cause lies in many ground truth lesion pixels classified as healthy skin (leading to high FN) or many ground truth healthy skin pixels classified as lesion (leading to high FP).

60 (768 by 576 pixel) images of melanocytic lesions were acquired using a Fotofinder digital dermatoscope. 12 copies of each image where then printed on 13 cm by 18 cm photographic paper. A copy of each image together with a marker was given to each of 4 "junior", 4 "senior" and 4 "expert" dermatologists (respectively less that 1 year

of dermoscopy training, between 1 and 5 years, and more than 5 years). Each dermatologist was then asked to independently draw the border of each lesion with the marker. The images (and borders) were scanned and realigned to the same frame of reference. Finally, the contours provided by the markers were extracted and compared. This allowed the identification, for each pixel of each original image, of the set of dermatologists classifying it as part of the lesion proper or of the surrounding, healthy skin.

This approach required a considerable amount of engineering effort compared to that of similar studies in the literature. [27] had dermatologists use Adobe PhotoShop's "pencil" tool to draw a polygonal approximation of the contour. [14] and [12] had dermatologists indentify a sparse set of points in the contour and then fit the points to a second-order B-spline. [33] had dermatologists draw the border on a tablet computer. Our goal was to maximize the comfort of dermatologists, thus minimizing the noise in border localization caused by the use of unfamiliar drawing tools.

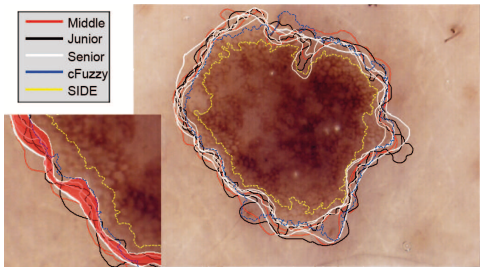


Figure 6.16: **Hand traced borders compared with c-fuzzy and SIDE**.*12 hand traced borders (red, black and white) and the shape detection obtained with modified c-Fuzzy (blue) and SIDE(yellow)*

Each of the 4 possible sets of 3 expert dermatologists was used to provide a "ground truth" from which the divergence of the remaining expert dermatologist, of the 4 senior and the 4 junior dermatologists, as well as of 4 segmentation algorithms was assessed. Figure 6.17 shows the average value (over the 60 images and the 3 ground truth segmentations) of the values of $\frac{FP}{TP+FP}$ and of $\frac{FN}{TP+FP}$.

The 4 algorithms are representative of the 3 main classes of automated lesion segmentation techniques in the literature, as well as of a fourth, novel technique.

The first class uses edges and smoothness constraints to identify the lesion. We implemented GVF Snakes [24]: a promising approach, though with a number of serious shortcomings. The algorithm requires a good initial segmentation to converge, a preprocessing such as black frame removal or hair removal [13, 25], and a morphological postprocessing to refine the results.

The second class performs color clustering directly on the image: this includes Modified JSEG [12] and SIDE [26]. We implemented the latter.

The third class performs clustering on the color histogram and then maps back to the original image. Mean-Shift [39] and Fuzzy c-means [47] are representative of

	Ground Truth, average of: **Experts** 2, 3, 4		**Experts** 1, 3, 4		**Experts** 1, 2, 4		**Experts** 1, 2, 3	
	$\frac{FN}{TP+FP}$	$\frac{FP}{TP+FP}$	$\frac{FN}{TP+FP}$	$\frac{FP}{TP+FP}$	$\frac{FN}{TP+FP}$	$\frac{FP}{TP+FP}$	$\frac{FN}{TP+FP}$	$\frac{FP}{TP+FP}$
Cal.Expert	2.51%	11.10%	10.18%	3.07%	9.10%	3.56%	5.21%	5.70%
Senior 1	1.50%	10.58%	3.15%	8.41%	3.06%	8.74%	2.85%	9.72%
Senior 2	1.60%	13.23%	2.66%	10.71%	2.65%	11.04%	2.39%	12.00%
Senior 3	5.19%	7.47%	7.29%	5.64%	7.01%	5.80%	6.67%	6.68%
Senior 4	6.27%	6.61%	8.84%	5.07%	8.58%	5.26%	8.10%	6.06%
Junior 1	1.96%	14.37%	3.15%	11.90%	3.05%	12.23%	2.91%	13.25%
Junior 2	0.85%	14.51%	2.19%	12.17%	2.15%	12.52%	1.99%	13.53%
Junior 3	1.01%	17.63%	1.80%	14.96%	1.87%	15.32%	1.68%	16.34%
Junior 4	1.16%	11.02%	3.22%	8.88%	3.14%	9.21%	2.94%	10.19%
c-Fuzzy	5.34%	5.62%	8.21%	4.28%	8.11%	4.69%	7.32%	5.28%
SIDE	23.58%	2.88%	28.46%	2.12%	28.33%	2.31%	26.83%	2.69%
Stat. Thre.	26.39%	11.29%	24.84%	11.17%	25.23%	12.31%	22.80%	12.23%
Snakes	10.03%	19.36%	12.27%	17.66%	12.47%	18.02%	11.46%	18.62%

Figure 6.17: **A Touring test approach**. *Average divergence of each expert dermatologist from the ground truth provided by the other three; and average divergence of senior and junior dermatologists and of* 4 *segmentation algorithms from the same ground truth. Divergence is measured as false negative area (FN: lesion pixels misclassified as healthy skin) and false positive area (FP: healthy skin pixels misclassified as lesion) as a percentage of the proposed segmentation area (TP+FP: pixels corretly or incorrectly classified as lesion)*

this class. These clustering algorithms work either using the RGB space [13], the B component [33], the Lab space [62], or the Pricipal Component decomposition [47].

A technique that does not fit into any of the above could be based on *Statistical Thresholding* - in a nutshell, classifying as lesion those portions of skin that statistically differ in color from healthy skin. Given the average RGB color μ and matrix variance Σ of an healthy patch of skin (e.g. taken from the boundaries of the image) each pixel is classifies as lesion according to

$$d(c,\mu) \geq k \cdot |\Sigma|$$

where d is the Euclidean distance in the color space and k is a scalar controlling the sensitivity of the algorithm. Obviously, the algorithm does not perform well on lesions covering only a small region of the image: this is a problem common to many algorithms that can be easily fixed with a crop of the image frame. The advantages of this approach are that it is simple to implement, and that it corresponds to a very "natural" definition of lesion (as the portion of skin exhibiting sufficient color variance from healthy skin).

The results of Figure 6.17 show appreciable variability in the localization of the border of melanocytic lesions between human dermatologists. Even an expert dermatologist "misclassifies" (compared to a ground truth provided by other expert dermatologists) a portion of the image with an area between 2.2% and 39.1% of the area of

the lesion itself. Less experienced dermatologists have an even lower agreement with their expert colleagues: the misclassified portion of the image has an area between 7.4% and 62.5% of the area of the lesion itself for "senior" and between 5.9% and 152.4% for "junior" dermatologists.

Although not entirely apparent from Figure 6.17, this divergence is not due to a systematic bias of individual dermatologists towards "tighter" or "looser" borders: we ranked all dermatologists for each lesion in order of increasing surface classified as lesion, and each dermatologist ranked first on at least one lesion, and at eighth or "larger" on at least another. On the other hand, expert dermatologists do show a very slight bias towards "tighter" borders (perhaps a symptom of greater confidence), and also, as should be expected, a somewhat greater agreement with other expert dermatologists than with less experienced ones.

It would certainly be interesting to study the impact of such variability on large, multi-operator epidemiological studies. These results seem to roughly confirm those of [27], though they are not directly comparable due to the different methodology ([27] evaluates the segmentation divergence of human dermatologists from a mix of human and algorithmic segmentations, rather than only from human segmentations). They also suggest that dermatoscopy skills require at least several years of training to mature.

In terms of algorithms, SIDE, Snakes and Statistical Thresholding did not perform very well, misclassifying a portion of the image with an area respectively between 8.4% and 92.6%, between 12.1% and 245.5%, and between 13.8% and 151.9% of the area of the lesion itself. These 3 algorithms were outperformed on average by every dermatologist, including ones belonging to the least experienced, "junior" cohort. As for Snakes, this might have been expected. As for Statistical Thresholding, this shows that unfortunately the most natural, axiomatic definition of lesion (as the portion of skin exhibiting sufficient color variance from healthy skin) fails to provide results that, in practice, match the actual intuition of the human eye. As for SIDE, its poor performance is somewhat unexpected, given the results of [26]. This may be due, in part, to the fact that SIDE is a particularly difficult algorithm to calibrate correctly - its performance could perhaps be improved with better fine-tuning than what we managed to achieve.

On the other hand, our variant of Fuzzy *c*-means performed extremely well. On average, it misclassified a portion of the image with an area between 3.7% and 50.2% of the area of the lesion itself (again, using as ground truths the segmentations provided by teams of three expert dermatologists). This is barely worse, and in 1 case out of 4 better, than the performance of the fourth, expert dermatologist used as "control" in each case. It is also significantly better than the performance of all remaining senior and junior dermatologists. Figure 6.16 provides a visual intuition of the quality of the results of this algorithm. Fuzzy *c*-means thus appears an excellent candidate to provide standardized, objective and highly reproducible segmentation of melanocytic lesions and assessment of corresponding features that closely match those of the most experienced dermatologists.

6.5 The cardial patch

Cardial patches pulsate at regular intervals. This spontaneous behavior is not forced by any chemical or electrical stimulation: provided an healthy environment, the patch "knows" it is made of miocardial cells and thus behave according to. We analyzed the pulsation (here defined as the variation of the area over time) obtaining an electrocardiogram-like plot. Our analysis directly measures mechanical movements such as inflation/deflation, opposed to ECG which infers mechanical movements by detecting small electrical activities. In details, we measured the area of the region bounded by the Active Contour (recall section 3.4.2) and filtered the obtained data with a median filter to denoise the signal. For non technicians, we developed a graphical user interface. In this context, we claim our tool to be less invasive than ECG, less expensive and in general more reliable because of the contactless interface with the patch.

Interestingly, some patches present an asymmetric V-shape pulsation, characterized by a little "step" at half the ascent. We are still speculating about the nature of this, wondering wheter is benignant or malignant. In any case this teaches that such peculiar waveforms could be coded and that the analysis of the trace could provide a quality assessment. The quality check may be performed by a human or directly by a machine vision system.

7 Synthesis

This chapter closes the work discussing the concept of *Synthesis*. Section 7.1 explains what we mean for Synthesis, while section 7.2 shows a practical model for the drosophila melanogaster epithelium.

7.1 Why synthesis

The synthesis step involves the creation of a **model** of the structure of interest. Observed data and prior knowledge are the guidelines in the design process. Models serve a number of purposes, for instance to predict a behavior, to mathematically analyze or to prove properties. Generally speaking, one model is often build to address a subset of all the possible purposes; for instance FEM (Finite Elements Methods) models analyze mechanical properties, omitting chemical reactions or magnetic fields.

7.2 A (drosophila) epithelium mechanical model

This section is mainly due to Assistant Professor Alessandro Abate, who started working on the model in 2006 and developed the first working code.

7.2.1 Goals of the model

The morphogenesis of the Drosophila Melanogaster fruit fly is well studied in biology, still some part of the developmental stage are missing. In particular, the formation of the grooves in the embryo is unexplained. 3D confocal analysis qualitatively suggests that grooves morphogenesis critically depends on mechanical forces applied to the epithelium and to the mesoderm. There are a number of distinct forces that seems to play a role in the formation of the grooves, such as *actin cables* and *myosin accumulation*. The main purpose of the model is thus to understand, analytically or with

simulations, the distribution and the extent of these forces, as well to infer the mechanical characteristics on the epithelium. The model should also gather new insight and drive new hypotheses.

7.2.2 A package of cells

The epithelium is modeled as a 2D network of cells. For simplicity, each cell was considered to be exagonal, as this is the most frequent shape in the drosophila epithelium. Cells are regularly packed, and they do not split or move relatively to the neighbor cells. The tissue is flat, but it is folded 3-dimensionally.

Each cell is assumed to be elastic. This is modeled as a mixture of springs and dampers. Among different possible configurations, we modeled the edges of the cell as a spring and a dumper in parallel [5]. We also added internal springs, connecting opposite vertices, to prevent shrinking and to enforce a convex shape. Springs and dampers have the usual number of parameters, such as maximal elongation, stiffness, resting length and friction (see figure 7.1).

The whole movement of the epithelium is obtained applying forces to the vertices of the cells and integrating them over time. The model we get is very close to a FEM, whereas we choose an explicit eulerian integration scheme rather than an implicit one. For small time steps, the difference between the two schemes is negligible.

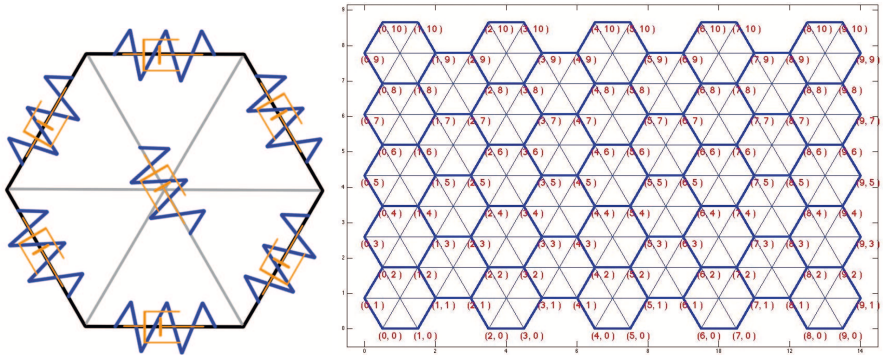


Figure 7.1: **Mechanical model of the Drosophila Melanogaster epithelium**. *The elasticity of the cell (left) is obtained by means of a network of springs and dampers. Each edge of the hexagon is a pair spring-damper in parallel. To prevent collapse and to enforce convexity, the model adds 3 virtual edges connecting opposite vertices. Again, each of these edges consists of a spring and a damper. The epithelium (right) is then modeled as a package of cells, regularly distributed in the space*

Referring to cloth simulation [5], in the following we'll call *particles r* the vertices of the cells, and *spring-damper* the edges. Every particle is described by a set of numbers:

- position p

- velocity **v**
- acceleration **a**
- mass m
- force **f**

We first compute the forces for every particle r_i, considering external forces, springs and dampers, according to

$$\begin{aligned}\mathbf{f}_i = &- \sum_{r_j neigh(r_i)} k_s \cdot (l_{ij} - ||\mathbf{p}_i - \mathbf{p}_j||)\mathbf{e}_{ij} \\ &- \sum_{r_j neigh(r_i)} k_d \cdot (\mathbf{v}_i - \mathbf{v}_j) + \mathbf{f}_{ext}\end{aligned}$$

where l_{ij} is the resting length of the spring connecting r_i and r_j, $\mathbf{e}_{ij}$ is the versor of the edge and $\mathbf{f}_{ext}$ takes in account all the external forces.

Once weve computed all of the forces in the system, we integrate them over time.

$$\begin{bmatrix} \dot{\mathbf{p}}_i \\ \dot{\mathbf{v}}_i \end{bmatrix} = \begin{bmatrix} \mathbf{v}_i \\ \mathbf{f}_i/m \end{bmatrix} \tag{7.1}$$

As mentioned, our discretization scheme is a standard forward Euler, which produces good results with small Δ_t

$$\begin{bmatrix} \mathbf{p}_i(t+1) \\ \mathbf{v}_i(t+1) \end{bmatrix} = \begin{bmatrix} \mathbf{p}_i(t) \\ \mathbf{v}_i(t) \end{bmatrix} + \begin{bmatrix} \mathbf{v}_i(t) \\ \mathbf{f}_i(t)/m \end{bmatrix} \cdot \Delta_t \tag{7.2}$$

7.2.3 Simulation results, parameter identification

We tested our model running several simulations, varying the numbers of cells, parameters and forces. The results are qualitatively similar to what commonly observed via microscope, although some works still need to be done. The roadmap towards a full comprehension of the morphogenesis mechanism is quite long, even if the mechanical model seems to be satisfactory. Our efforts lead to the hypothesis of two main external forces driving the formation of the groves: a *dorsal closure* force and an *actin cable* one. The former accounts for the closure of the initially flat patch of cells into a folded, cylindric, one; the latter produces the groves deeping them into the fruit fly body.

An open area of research is the identification of the real value of the parameters of the model. This is possible applying the model to the real data. The complete workflow is explained in figure 7.2: we first obtain the reticular shape from the video sequence using the techniques in chapters 5 and 4. Then, we correlate the frames using the Point Set Matching algorithm (as described in) to capture the evolution of the cells over time. If the parameters and the forces are correct, the time evolution of the synthetic cells will match that of the real cells.

Preliminary results are shown in figure 7.3.

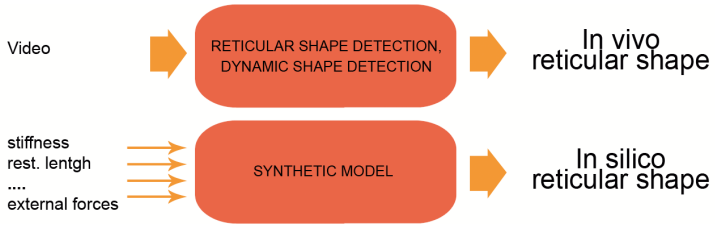


Figure 7.2: **Towards parameter identification**. *The mechanical models has a number of internal parameters, such as stiffness coefficients, friction, external forces. The exact value of these numbers leads to a simulation (*in silico *shape) equals to the shape captured from the video sequence (*in vivo *shape). In general, the problem is ill posed, and the solutions lie in a large subspace. Nonetheless, an analytical representation of the space, or even a numerical approximation, would lead significant insights in the morphogenesis*

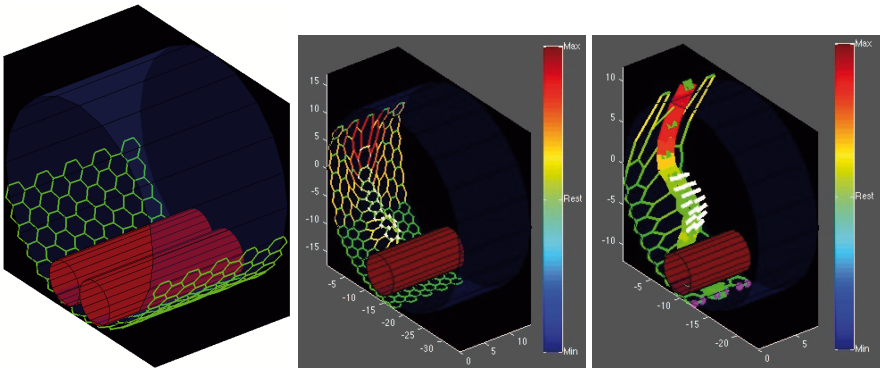


Figure 7.3: **Simulations of the grooves formation**. *Several simulations, run with a different number of cells and parameters, lead to the same qualitative results. Dorsal closure forces and actin cable forces act here a main role in the formation of the grooves. In white: actin cable force, pushing cells in the inside of the drosophila fruit fly; in green: dorsal closure forces, folding the plain patch into a cylinder*

8 ToolBox

The chapter describes the toolbox we developed to handle Generalized Active Contours, Random Walk Agents and J-maps. We use here a compact and readable pseudo-code even though the original source code was implemented in Matlab (Mathworks Inc.)

8.1 Generalized Active Contour toolbox

There are two types of implementation for snake models: the implicit one and the explicit one. Both implementations rely on a discretization of the continous curve $\mathcal{C}(s)$ both in space and time.

Implicit models, such as the formulation used in [42], embed the snake as the zero level set of a higher dimensional function and solve the corresponding equation of motion for a fixed time step. As briefly discussed in section 1.3, the boundary of the Active Contour $\mathcal{C}(s)$ is then recovered by $\varphi = 0$, where φ is a discrete-domain function. Such methodology is suited for the recovery of objects with complex shapes and unknown topologies. However, due the higher dimensional formulation, implicit models are unconvenient for shape analysis, visualization, and user interaction.

In explicit implementations, on the contrary, the representation of the boundary is explicitly stored in a variable for easy access and $\mathcal{C}(s)$ is approximate using a set of (ordered) points $C = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_n\}$ joined by a straight line. More sophisticated interpolation schemes are possible, such as quadratic or cubic bezier curves.

In other words, implicit models use a Lagrangian representation, whereas explicit models use an Eulerian one.

Each point has a mass, and forces -directly applied to the points- move them in space. Code snippet 1 shows the main loop of the explicit model provided within the toolbox. The loop is here intended to work on a static image `I` as presented in chapter 3. Line 1 and 2 initialize the mass and velocity vectors: `m` and `v` are respectively arrays containing the masses of the points (here set to unitary) and the velocity vectors (at the beginning set to zero). Line 3 initializes the position of the points `x`: in our experi-

Code snippet 1 - Generalized Active Contour main loop

```
1    m ← 1                          //mass vector
2    v ← 0                          //velocity vector
3    x ← initPosition()             //points initial position
4    I ← extractImage()
5    for t=1 to end {
6         f ← computeForces(I,x)          //forces acting on the boundary
7         a ← f/m
8         x,v ← timeIntegration(x,v,a) }  //move points according to the
                                          //preferred integration scheme
```

ments we chose a circular rest position. Indeed, the initial position of the points is not a critical factor and almost any other configuration is suitable. This is especially true for the Generalized Active Contour, which by its nature tends to define the energy term $\mathcal{E}$ as a convex function (recall equations 3.4 and 3.10). Line 6 and line 8 are the core of the method. **`computeForces(...)`** computes and outputs the forces acting on the Contour (acting on the *points of the Contour*), while **`timeIntegration(...)`** advances points in time using "a certain time-integration scheme".

The toolbox provides code for an explicit time integration. Provided that **`dt`** $\to 0$, the scheme is stable. More accurate schemes, such as Runge-Kutta or implicit ones [9], are still possible. If the simulation reveals unstable, one can force a smaller `dt` or use a damped integration scheme. A damped scheme dissipates kinetic energy, resulting in a (usually) non-oscillating behavior. Snippet 2 shows the damped scheme provided in the toolbox, in which the velocity `v` is totally dissipated at every time step (line 2). In our simulation we almost always used the non-damped **timeIntegration(.)**.

Code snippet 2 - Forward damped time integration

```
1    function dampedTimeIntegration(x,v,a)
2         v ← 0
3         return timeIntegration(x,v,a)
```

The **`computeForces(I,x)`** function is the core function. It defines the energy (equation 3.10) of the (Generalized) Active Contour and computes the forces acting on points as derivative (equation 3.6). The function takes two input: the position of the points **`x`** and the image `I`. Even though the two inputs are often used together and simultaneously, one can argue that the former is primarly used to compute configuration-based energies (such as $\mathcal{S}_g(\mathcal{C})$) while the latter to compute image-based energies (such as $\mathcal{P}_g(\mathcal{C})$) (recall the distinction between *model energy* and *image energy* introduced in section 3.3).

Code snippet 3 shows a straightforward implementation, derived directly from equation 3.12. Lines 5 – 7 compute the classical Active Contour forces according to

Code snippet 3 - Generalized Active Contour Force computation

```
1   function computeForces(I,x)
2       f ← 0
2       e ← 0
3       for i=1 to |x| {
4           //Active Contours forces
5           f[i] ← firstDerivativeF(x)              //∂C/∂s
6           f[i] ← f[i] + secondDerivativeF(x)      //∂²C/∂s²
7           f[i] ← f[i] + imageDerivativeF(I)       //∇_D I
8           //Generalized Active Contours forces
9           f[i] ← f[i] + numericalDerivative(E1(I,x))
10          f[i] ← f[i] + numericalDerivative(E2(I,x))
11              ...             ...
12          f[i] ← f[i] + numericalDerivative(E_{ng+nf}(I,x)) }
13      return f
```

equation 3.8:

$$F(s) = -\nabla_{\mathcal{D}}[I] + 2\frac{\partial}{\partial s}\left(\alpha(s)\frac{\partial \mathcal{C}}{\partial s}\right) + 2\frac{\partial^2}{\partial s^2}\left(\beta(s)\frac{\partial^2 \mathcal{C}}{\partial s^2}\right) \tag{8.1}$$

This a well-known task [7, 48].

Lines 9 – 12 show the computation of the generalized forces, in the pseudocode calculated as numerical derivative (see the variational principle, equations 3.6 and 3.7) of the generalized energies $\mathcal{E}_1, \ldots, \mathcal{E}_{n_g+n_f}$. For details on the **numericalDerivative()**, please look at the very good [41], [53].

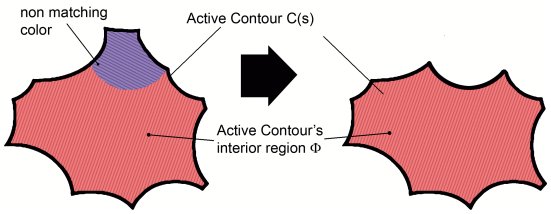
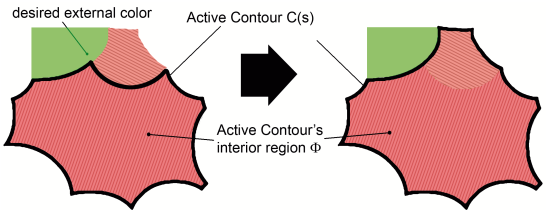
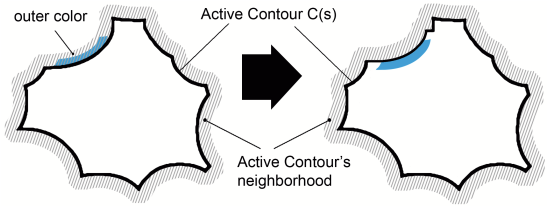
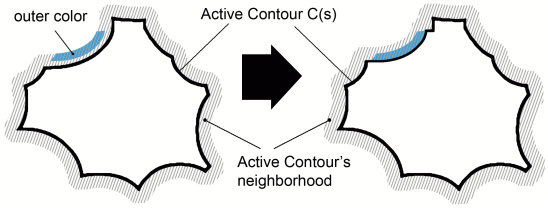
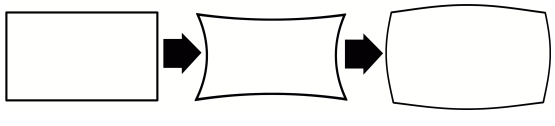
8.1.1 The "Energy" toolbox

Many efforts have been directed towards the creation of a toolbox of energy functions. The idea is to use these functions as building blocks (code snippet 3) for a more general purpose algorithm.

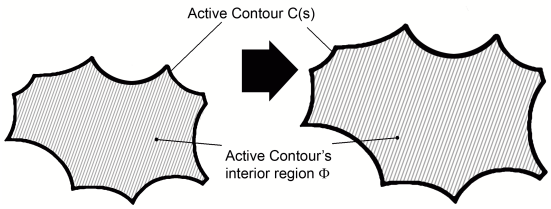
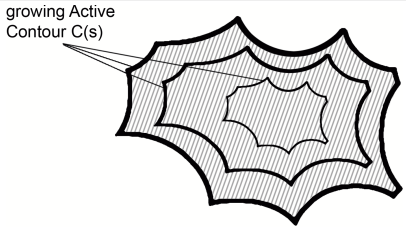
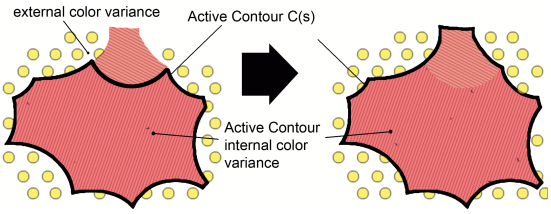
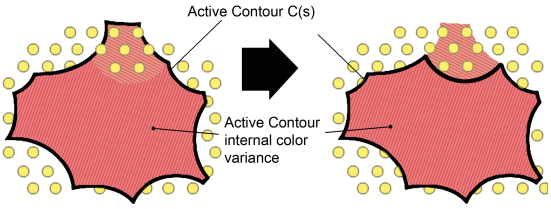
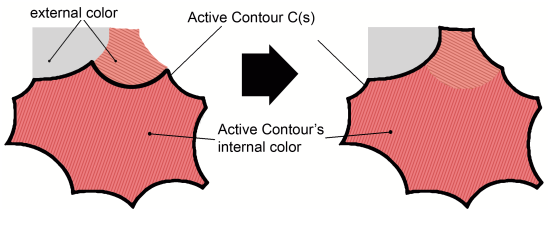
Tables I and II show a number of functionals provided in the toolbox [1]. Our design choice was to mainly use linear and gaussian-based energies. We found this to be a good tradeoff between complexity and robustness. Indeed, if the real density is known, one can design more accurate terms.

[1] To keep notation simple, we here used the symbol Σ to compute the variance of a distribution - usually color-. We also introduced the trivial function "Area" to compute the area of a Contour

Energy Toolbox I/II

Name	Figure	Definition
Inner Color()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \int_{C^-} \lvert I(\omega) - c \rvert \, d\omega$ c =desired inner color C^- = interior of C
Inner Color energy drives the Contour to have an internal color c		
Outer Color()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \int_{C^+} \lvert I(\omega) - c \rvert \, d\omega$ c =desired inner color C^+ = exterior of C
Outer Color energy drives the Contour towards an external color c The areas of c color remain out of the Contour		
Surrounding Inner Color()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \int_{C^+} \lvert I(\omega) - c \rvert \, d\omega$ c =desired outer color
Surrounding Inner Color forces the Contour to have a small internal neighbor of color c		
Surrounding Outer Color()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \int_{C^+} \lvert I(\omega) - c \rvert \, d\omega$ c =desired outer color
Surrounding Outer Color forces the Contour to have a small external neighbor of color c The areas of c color remain out of the Contour		
Spikeness()		$E = -\frac{min(\mu,\mu_{max})}{\mathcal{N}}$ $\mu = \int_C \kappa(s) ds$ $\kappa(s) = \frac{\partial^2 C(s)}{\partial s^2}$ μ_{max} =max value for μ $\mathcal{N}$ =normalizing constant
Depending on the sign, force a spiky appearance or a cloudy one c		

Energy Toolbox II/II

Name	Figure	Definition
Area()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \int_{C^-} \lvert area(C) - c \rvert \, d\omega$ c =desired area size
Area energy drives the Contour towards an area size c		
Enflate() Deflate()		$E = -\frac{min(\mu,\mu_{max})}{\mathcal{N}}$ $\mu = \int_{C^-} ds$ μ_{max} =max value for μ $\mathcal{N}$ =normalizing constant
Enflate (or Deflate) the Active Contours forcing a larger (smaller) area. μ_{max} is an upper bound, possibly unreachable		
Inter Class Color Variance()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \lvert \Sigma(I(C^+)) - \Sigma(I(C^-)) \rvert$
Inter Class Color Variance enhances the color variance between the interior and the exterior		
Internal Minimal (Maximal) Color Variance()		$E = -\frac{min(\mu,\mu_{max})}{\mathcal{N}}$ $\mu = \Sigma(I(C^-))$ μ_{max} =max value for μ $\mathcal{N}$ =normalizing constant
Internal Minimal (or Maximal) Variance favours a small (large) internal color variance		
Inter Class Color()		$E = -\frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{\mu^2}{2\sigma^2}}$ $\mu = \lvert \int_{C^-} I(C) d\omega +$ $- \int_{C^+} I(C) d\omega \rvert$
Inter Class Color enhances the color difference between the interior and the exterior		

As a general guideline, we used a gaussian function to impose a specific value on a specific metric. For instance, the **`Inner Color()`** energy term drives the Contour towards a c interior color. All the deviations from c are discouraged using a gaussian weight.

The linear functions are instead used if a specific "equilibrium" value is not desired or one wants for a metric to grow as much as possible. For instance, the **`Spikeness()`** energy term encourages the Contour to grow in spikeness regardless of its value.

8.2 Random Walk Agents toolbox

Recalling chapter 4, the Random Walk Agents are a suitable approach to detect reticular shapes. The idea is to run a number of agents on the "digital landscape" (the image I), each of them following a path. The paths traced by the Agents generate a graph structure drawn over the real (reticular) one.

Code snippet 4 shows the main loop of the approach. There are a number of

Code snippet 4 - Randow Walk Agents main loop

```
1    PQ ← initPriorityQueue()      //create a priority queue
2    G ← initGraph()               //create an empty graph
3    while PQ not empty() {
4         A ← dequeue(PQ)          //extract the best agent
5         valid, border ← validateAgent(A,I,G)
6         if valid
7              G← add2Graph(G,A) //add to the graph
8         if (valid and not(border)) {
9              E ← computeEnergyFunction(I,A_i)
10             D ← pickDirections(E)
11             for k=1 to |D| {
12                  A_k ← moveAgent(A,D_k)
13                  PQ ← enqueue(A_k,PQ) }
14        }
15   }
```

elements playing a role. Line 1 and 2 respectively create a priority queue **`PQ`** with one Agent and an empty graph **`G`**. The former is used to list the moving Agents; the latter to store the positions traced by the Agents and their paths. As discussed in section 1.3, the shape representation model is a graph augmented with the geometrical positions of the nodes. At the end of the procedure the shape will be given by the nodes and the edges of the graph.

The priority queue provided in the Toolbox is based on a Heap Tree [19]. Lines 4 and 13 show the role of the priority queue: after an Agent moves, it gets stored in the

queue. The queue is ranked by increasing values of the **E** function, thus at every cycle only the best Agent is choosen for moving.

After an Agent gets dequeued, the **validateAgent()** function tests for validity. The Agents must obey a number of rules to be "valid": for instance a valid agent does not create small loops nor is too close to an existing node in **G**. If the Agent is valid, its position is inserted into the graph **G**: **add2Graph()** takes care of the insertion, creating the right edges. If the agent is valid and is not on the border, it also moves (line 12). The directions of advancement are given by **computeEnergyFunction()** and **pickDirections()** (lines 9 – 10), respectively computing the energy function $\hat{\mathcal{E}}$ (equation 4.4) and the heading directions Θ (equation 4.5). Finally, **moveAgent()** implements the motion equation 4.1.

The procedure iterates until the queue is empty, i.e. no more Agents are on the frame.

Code snippet 5 shows the **validateAgent()** function

Code snippet 5 - valid Agent

```
1    function validateAgent(I,G,A)
2         G'← add2Graph(G,A)
3         valid ← true
4         if (loopLength(G')<10)
5              valid ← false
6         border ← closeToBorder(A,I)
7         return valid,border
```

loopLength() and **closeToBorder()** test for geometrical condition. The former tests the graph **G'** to contain a loop smaller than 10 edges [19], the latter for the position of the Agent **A** to be close to the border of the image **I**. Code snippet 6

Code snippet 6 - compute the Energy

```
1    function computeEnergyFunction(I,A)
2         E← malloc(360)              //allocate memory for E
3         for theta=0 to 359 {
4              Ω ← createSector(A,theta)
5              E[theta] ← sum(I - Ω }      //integral 4.4
6         return E
```

computes the energy $\hat{\mathcal{E}}$. Line 4 creates a mask centered on the Agent $\mathcal{A}$ [55], line 5 computes the integral (here a discrete sum) of equation 4.4.

8.3 J-Maps toolbox

J-Maps were introduced in section 8.3. Briefly recalling to memory, they are a tool to morph a shape Φ_t into Φ_{t+1}, where Φ_t is the shape (detected) on frame I_t and Φ_{t+1} is the shape on frame I_{t+1}. They are mostly used in the Active Contour incremental approach (section 5.3).

Given two frames I_t, I_{t+1} and a location `x=x,y` on I_t, code snippet 5 shows the creation of a single J-Map. The procedure first allocates the memory to store **J** (line

Code snippet 7 - J-Map computation

```
1    function getJMap(I_t,I_{t+1},y,x)
2    N ← 20                          //size of the map
3    J ← malloc(2N+1,2N+1)           //allocate memory
4    for ty=-N to N {
5    for tx=-N to N {                //scan all the J-Map
6         if (tx²+ty²>N²
7              J[ty,tx] ← ∞
8         else {
9              for xx=-N to N {
10             for yy=-N to N {      //likelihood of being the right translation
11                  J[ty,tx] ← J[ty,tx]+
                         + |I_t[y+yy,x+xx]-I_{t+1}[y+ty+yy,x+tx+xx]| } }
12             }
13        } }
14   return J
```

3), then fills the entries of the map in a doubly-nested loop (lines 4 – 5). The logical meaning of `ty` and `tx` is "guessed shift": the procedure pretends the original point `y,x` to be shifted on `y+ty,x+tx` and evaluate the likelihood of this on lines 9 – 11.

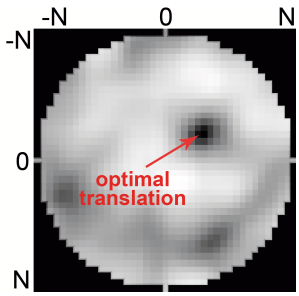


Figure 8.1: **J-map**. *The output of* **`getJMap()`** *is a single J-Map. Each point of the map compute the likelihood of point* `y,x` *on* I_t *to be mapped on* `y+ty,x+tx` *of* I_{t+1}

The computation of the all J-Maps iteratively calls **`getJMaps()`** on the points

of the Contour $\mathcal{C}_t$.

Code snippet 8 - J-Maps computation

```
1    function getJMaps(I_t, I_{t+1}, C_t)
2    Ja ← malloc(.,.,|C_t|)      //allocate the right amount of memory
3    for i=1 to |C_t| {          //for each point of the Contour
4         y,x ← C_t[i]
5         Ja[i] ← getJMap(I_t, I_{t+1}, y, x) }
6    return Ja
```

8.3.1 J-Maps correction

As discussed in section 5.3.1, often J-Maps need to be corrected. Code snippets 9 and 10, based on equation 5.18, implements such a correction. The `while` loop is

Code snippet 9 - J-Maps computation

```
1    function correctJMaps(Ja)
2         loop ← 0
3         while (true) {
4              loop ← loop+1
5              Jac ← onePassCorrection(Ja)
6              noChanges ← numberOfChanges(Jac,Ja)
7              Ja ← Jac
8              if (noChanges==0 or loop>5)
9                   exitLoop
10        }
11        if (loop>5)
12             return 'fail'
13        return Ja
```

built around the **`onePassCorrection()`** function. If no more corrections happens, then the J-Maps are adjusted. **`numberOfChanges(Jac,Ja)`** is a suitable function that counts the number of global minima locations undergoing a change. If `noChanges` is equal to zero, then **`onePassCorrection()`** had no effect on the maps and hence the while loop terminates. Line 11 tests for the number of loops of the statement. If the number is greater than 5 no convergence has been reached, and the procedure is manually marked as 'failed'. As we observed in our experiments, a typical convergence involves two or three loops. This kind of check is necessary. In

fact, large deformations and/or abrupt changes of the structure potentially lead to a miscalculation of the optical flow. This is a violation of equation 5.4.

Code snippet 10 - J-Maps one pass correction

```
1   function onePassCorrection(Ja)
2       Jac ← malloc(...)
3       for i=0 to |Ja| {
4           Jnn ← getCloseJMaps(i,Ja)
5           v̄y,v̄x,σ,← getGaussParameter(Jnn)
6           Jac[i] ← Ja[i] · gaussWin(v̄y,v̄x,σ) }
7       return Jac
```

Snippet 10 shows the details of the **onePassCorrection()** function. It takes as input the whole set of maps to be corrected (**Ja**), and outputs the adjusted version (**Jas**). Given a single J-Map "i", the function **getCloseJMaps()** on line 4 returns all the J-Maps that are close in space to the "i-th". Recalling the meaning of a J-Map, each J-Map is associated to a point of the Contour $\mathcal{C}$, so in the end a map is univocally associated to a precise location in space. **getCloseJMaps()** returns all the J-Maps that refers to points close to the point referred by the "i-th". The subscript "nn" means "nearest-neighbor", as the function implements a sort of nearest neighbor search based on kd-trees (see the excellent [30]).

getGaussParameter(Jnn) takes all the mentioned maps and computes the parameter of the gaussian curve fitting the locations of the local minima of **Jnn**. Finally line 6 weights **Ja**[i] multiplying by the gaussian curve.

We repropose figure 5.3 (figure 8.2 below), showing the output of **correctJMaps()** on a single J-map.

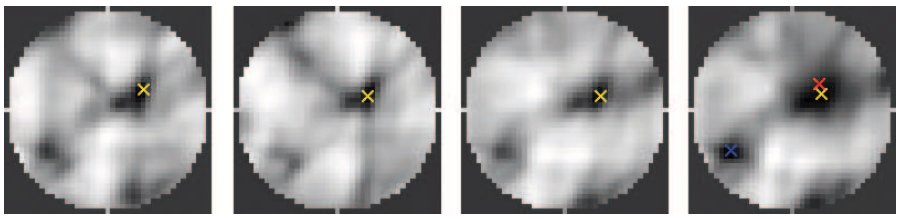


Figure 8.2: **Correction of the J-Maps**. *The J-map on the right, $\bar{J}_s$, is computed from J_s according to equation 5.18. The blu cross is the position of the old global minimum. The red cross is the center of the gaussian $(\bar{v}_x, \bar{v}_y)$. The yellow cross is the location of the new global minimum*

9

Conclusions and Future Works

9.1 Conclusions

This work presented an energy-based unified framework for shape detection, developing a theoretical background and then applying it to real cases. Shape detection is the first and most important step to build a robust computer vision system: the work emphasized theoretical aspects, but also showed a number of applications. In particular, the developed algorithms proved to be effective in a number of case studies, both in the static and in the dynamic context.

Shape analysis can then be considered (one of) the "goal" of the shape detection. A chapter is thus dedicated to such topic, addressing popular problems in the community. Having a shape is in fact useless if usefuls facts cannot be extraced.

Shape synthesis is a step further. Synthesis comes when the knowledge is so high that one can guess the hidden nature of the object and model it. The model can match *in silico* the *in vivo* behavior and thus be used to predict events.

9.2 Future works

Much has still to be done. Every chapter ends in an opened way. The solution of a practical problem and the development of a theoretical background is the bare minimum to consider a real progress towards the full knowledge of a wide topic such Computer Vision and Image Analysis. The effort of this work is to develop tools and methodologies useful in a real-life context. For this reason, we rarely used synthetic test to prove effectiveness. The ambitious big picture is not to publish a good-looking book, with fancy pictures and formulas, rather to provide reliable tools. Much more ambitiously, we would like to use these same tools for several years, with little or no tuning, dealing with different problems.

Generalized Active Contours need to be refined. In particular the link between probabilistic-based energies and the physicality of the object needs to be better understood. Probabilistic density functions are just like black boxes, modeling "physical"

characteristics otherwise too complex to be used. The concept is similar to the cast of a dice. The concept of probability hides the complexity of the object, allowring for an easier treatment.

The Random Walk Agents are an interesting tool to detect reticular shapes. They are based on an analogy with the human vision system, that “follows” traces locally with small movements of the eyes and the head. The walk agents formalize this behavior, and tell us what “to follow” means.

The forecoming years will bring us more knowledge, and hopefully more problems will be solved.

List of Figures

Bibliography

[1] http://vasc.ri.cmu.edu/idb/html/motion/house/index.html.

[2] M. Adams, S. Celniker, and R. Holt. The genome sequence of drosophila melanogaster. *Science*, (5461):2185–2195, 2000.

[3] K. R. T. Aires, A. M. Santana, and A. A. D. Optical flow using color information: preliminary results. *Proceedings of the 2008 ACM symposium on Applied computing*, pages 1607–1611, 2008.

[4] K. W. B. Smolka. Random walk approach to image enhancement. *Signal Processing 81 (2001) 465-482*, 2001.

[5] D. Baraff and A. Witkin. Large steps in cloth simulation. *Proceedings of Annual Conference Series SIGGRAPH*, 1998.

[6] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(2), 1995.

[7] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, New York, USA, 1998.

[8] P. Brigger, J. Hoeg, and M. Unser. B-spline snakes: A flexible tool for parametric contour detection. *IEEE Transactions on Image Processing*, 9(9), 2000.

[9] J. C. Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*. John Wiley Sons Inc, 1986.

[10] M. Carcassoni and E. Hancock. Correspondence matching with modal clusters. *IEEE Transactions on Pattern Analalysis and Machine Intelligence*, 25(12):1609–1615, 2003.

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

[12] M. Celebi, Y. Aslandogan, and W. Stoecker. Unsupervised border detection in dermoscopy images. *Skin Res. and Tech,*, 13:454–462, 2007.

[13] M. Celebi, H. Kingravi, H. Iyatomi, J. Lee, Y. A. Aslandogan, W. V. Stoecker, R. Moss, J. M. Malters, and A. A. Marghoob. Border detection in dermoscopy images using statistical region merging. *SPIE*, 13, 2007.

[14] M. E. Celebi, G. Schaefer, and H. Iyatomi. Objective evaluation of methods for border detection in dermoscopy images. pages 3056–3059, Aug 2008.

[15] G. Chartrand. *Introduction to Graph Theory*. Dover Publications, 1984.

[16] G. Chartrand, Prindle, Weber, and Schmidt. *Graph as Mathematical Models*. Inc. Boston, 1977.

[17] G. Chartrand and P. Zhang. *Chromatic Graph Theory (Discrete Mathematics and Its Applications)*. Chapman and Hall, 2008.

[18] T. H. Cormen. *Algorithmic Complexity - CRC Handbook of Discrete and Combinatorial Mathematics*. CRC Press, 2000.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press and McGraw-Hill, 2001.

[20] L. Corporation. What is constructive solid geometry? Technical Report, 2006.

[21] S. Dimmeler, J. Burchfield, and al. Cell-based therapy of myocardial infarction. *Arterioscler. Thromb. Vasc. Biology*, 2007.

[22] J. Duncan and N. Ayache. Medical image analysis: Progress over two decades and the challenges ahead. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:85–106, 2000.

[23] N. A. Epidaure. A research project in medical image analysis, simulation, and robotics at inria. *Trans. on Medical Imaging*, 22(10):1185–1201, 2003.

[24] B. Erkol, R. H. Moss, R. J. Stanley, W. V. Stoecker, and E. Hvatum. Automatic lesion boundary detection in dermoscopy images using gradient vector flow snakes. *Skin Res. and Tech,*, 11:17–26, 2005.

[25] H. Z. et al. Feature-preserving artifact removal from dermoscopy images. *Proc. SPIE*, 2008.

[26] J. G. et al. Segmentation of dermatoscopic images by stabilized inversediffusion equations. *Proc. ICIP*, 3:823–827, 1998.

[27] J. G. et al. Validation of segmentation techniques for digital dermoscopy. *Skin Research Technology*, 8(4):240–249, 2002.

[28] D. J. Fleet and Y. Weiss. *Optical Flow Estimation - Handbook of Mathematical Models in Computer Vision*. Springer, 2006.

[29] L. Grady. Multilabel random walker image segmentation using prior models. *Proc. of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1(1), 2005.

[30] A. G. Gray and A. W. Moore. n-body problems in statical learning. *Advances in Neural Information Processing Systems*, 13(5):521–527, 2000.

[31] J. Guillod, P. Schmid-Saugeon, D. Guggisberg, J. P. Cerottini, R. Braun, J. Krischer, J. H. Saurat, and K. Murat. Validation of segmentation techniques for digital dermoscopy. *Skin Res. and Tech*, 8(4):240–249, November 2002.

[32] Z. Guopu, Z. Shuqun, C. Xijun, and C. Wang. Novel gradient vector flow-based balloon force for active contours. *Journal of Electronic Imaging*, 18(2), 2009.

[33] H. Iyatomi, H. Oka, M. Saito, A. Miyake, M. Kimoto, J. Yamagami, S. Kobayashi, A. Tanikawa, M. Hagiwara, K. Ogawa, G. Argenziano, H. P. Soyer, and M. Tanaka. Quantitative assessment of tumour extraction from dermoscopy images and evaluation of computer-based extraction methods for an automatic melanoma diagnostic system. *Melanoma Res.*, 16(2):17–26, 2005.

[34] M. Kass, A. Witkin, , and D. Terzopoulos. Snakes - active contour models. *International Journal of Computer Vision*, 1987.

[35] V. V. Kindratenko. On using functions to describe the shape. *Journal of Mathematical Imaging and Vision*, (18):225–245, 2003.

[36] M. A. Laflamme, C. E. Murry, and J. Mascaro. Regenerating the heart. *Nature BiotechArch.*, 23(7):845–856, 2005.

[37] D. Ma, K. Amonlirdviman, R. Raffard, A. Abate, C. Tomlin, and J. Axelrod. Cell packing influences planar cell polarity signaling. *The Proceedings of the National Academy of Sciences*, 105(48):18800–18805, December 2008.

[38] J. Mascaro. The dermatologist's position concerning nevi: a vision ranging from "the ugly duckling" to "little red riding hood". *Arch. Dermatol.*, 134(5461):14841485, 1998.

[39] R. Melli, C. Grana, and R. Cucchiara. Comparison of color clustering algorithms for segmentation of dermatological images. *Proc. of SPIE*, 2006.

[40] D. Mumford. Mathematical theories of shape: Do they model perception? *Proc. Geometric Methods in Computer Vision Conference, SPIE*, pages 2–10, 1991.

[41] R. L. Navin. *The Mathematics of Derivatives: Tools for Designing Numerical Algorithms*. Wiley, 2007.

[42] S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.

[43] M. Radisic and H. Park. From the cover: Functional assembly of engineered myocardium by electrical stimulation of cardiac myocytes cultured on scaffolds. *Proceedings of the National Academy of Sciences*, 101(52):18129–18134, 2004.

[44] M. H. Reisner and E. Rubin. Essentials of rubin's pathology. (5461):513, 2008.

[45] P. L. Rosin. Measuring shape: ellipticity, rectangularity, and triangularity. *Machine Vision and Applications*, (14):172–184, 2002.

[46] T. Schenhagen, C. Fink, and al. Three-dimensional reconstitution of embryonic cardiomyocytes in a collagen matrix: a new heart muscle model system. *FASEB J.*, 11(8):683–694, 1997.

[47] P. Schmid. Segmentation of digitized dermatoscopic images by two-dimensional color clustering comparison. *IEEE Trans. on Medical Imaging*, 18(2):164–171, 1999.

[48] J. A. Sethian. *Level Sets Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science*. Cambridge University Press, Cambridge, UK, 1999.

[49] L. Shapiro and J. Brady. Feature-based correspondence: An eigenvector approach. *Image and Vision Computing*, 10(5):283–288, June 1992.

[50] A. Silletti, A. Cenedese, and A. Abate. The emergent structure of the Drosophila wing: a dynamic model generator. In *Proceedings of the International Conference on Computer Vision, Theory and Applications (VISAPP 09)*, pages 406–410, Lisboa, PT, February 2009.

[51] S. H. Srinivasan and M. Kankanhalli. Wide baseline spectral matching. In *Proceedings of the 2003 International Conference on Multimedia and Expo (ICME 03)*, pages 93–96, Washington, DC, USA, 2003.

[52] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischert. Elastically deformable models. *International Conference on Computer Graphics and Interactive Techniques*, 21(4):205–214, 1987.

[53] W. T. Vetterling and B. P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, 2002.

[54] H. Wang and E. Hancock. Correspondence matching using kernel principal components analysis and label consistency constraints. *Pattern Recognition*, 39(6):1012–1025, 2006.

[55] A. Watt. *3D Computer Graphics*. Pearson Addison Wesley, 2000.

[56] C. Xiao and H. C. Nelson. Corner detector based on global and local curvature properties. *Optical Engineering*, 47(5):057008, 2008.

[57] X. Xie and M. Mirmehdi. Mac: Magnetostatic active contour model. *IEEE Transaction on Patter Analysis and Machine Intelligence*, 30(4):632–646, 2008.

[58] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *EEE Transactions on Image Processing*, 7(3):359–369, 1998.

[59] B. Y. Xun Hou, Wei Zhao. Edge detection and contour tracing of medical cell images. In *Proc. SPIE, Vol. 6279, 62792M (2007); DOI:10.1117/12.725262*, 2007.

[60] A. J. Yezzi and S. Soatto. Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images. *International Journal Computer Vision*, 53(2):153–167, 2003.

[61] A. L. Yuille, P. W. Hallinan, and D. S. Cohen. Feature extraction from faces using deformable templates. *Int. Journal Computer Vision*, 8(2):99–111, 1992.

[62] H. Zhou, M. Chen, L. Zou, R. Gass, L. Ferris, L. Drogowski, and J. M. Rehg. Spatially constrained segmentation of dermoscopy images. *Proc. 5th IEEE Int. Symp. on Biomed. Imag.*, pages 800–803, 2008.