Universidad de la República
Facultad de Ingeniería

Segmentation and polyp detection in virtual colonoscopy: a complete system for computer aided diagnosis

Marcelo Fiori

Dr. Pablo Musé, Dr. Guillermo Sapiro
Thesis Directors

Comitee of Examiners
Dr. Afra Zomorodian (Dartmouth College), Dr. Omar Gil (UdelaR)
Dr. Álvaro Pardo (UCUDAL), Dr. Gregory Randall (UdelaR)

Montevideo, Uruguay
2011

Resumen

El cáncer colorectal es una de las mayores causas de muerte por cáncer en el mundo. La detección temprana de pólipos es fundamental para su tratamiento, permitiendo alcanzar tasas del 90% de curabilidad. La técnica habitual para la detección de pólipos, debido a su elevada performance, es la colonoscopía óptica (técnica invasiva y extremadamente cara). A mediados de los '90 surge la técnica denominada colonoscopía virtual. Esta técnica consiste en la reconstrucción 3D del colon a partir de cortes de tomografía computada. Es por ende una técnica no invasiva, y relativamente barata, pero la cantidad de falsos positivos y falsos negativos producida por éstos métodos está muy por encima de los máximos aceptados en la práctica médica. Los avances recientes en las técnicas de imagenología parecerían hacer posible la reducción de estas tasas. Como consecuencia de esto, estamos asistiendo a un nuevo interés por la colonoscopía virtual.

En este trabajo se presenta un sistema completo de diagnóstico asistido por computadora. La primer etapa del sistema es la segmentación, que consiste en la reconstrucción 3D de la superficie del colon a partir del volumen tomográfico. El aporte principal en este paso es el suavizado de la imagen. A partir de la superficie, se detectan aquellas zonas candidatas de ser pólipos mediante una estrategia multi-escala que permite delinear con precisión la zona. Luego para cada candidato se extraen características geométricas y de textura, que son calculadas también en el tejido que rodea la zona a efectos de compararlas. Finalmente las zonas candidatas se clasifican utilizando SVM. Los resultados obtenidos son prometedores, permitiendo detectar un 100% de los pólipos mayores a 6mm de diámetro con menos de un falso positivo por caso. Por otro lado, el sistema es capaz de detectar el 100% de los pólipos mayores a $3mm$ de diámetro con 2.2 falsos positivos por caso.

Esta versión incluye cambios y agregados propuestos por los integrantes del tribunal, a quienes agradezco la lectura dedicada y los valiosos aportes y comentarios.

Abstract

Colorectal cancer is the second leading cause of cancer-related death in the United States, and the third cause worldwide. The early detection of polyps is fundamental, allowing to reduce mortality rates up to 90%. Nowadays, optical colonoscopy is the most used detection method due in part to its relative high performance.

Virtual Colonoscopy is a promising alternative technique that emerged in the 90's. It uses volumetric Computed Tomographic data of the cleansed and air-distended colon, and the examination is made by a specialist from the images in a computer. Therefore, this technique is less invasive and less expensive than optical colonoscopy, but up to now the false positive and false negative rates are above the accepted medical limits. Recent advances in imaging techniques have the potential to reduce these rates; consequently, we are currently re-experiencing an increasing interest in Virtual Colonoscopy.

In this work we propose a complete pipeline for a Computer-Aided Detection algorithm. The system starts with a novel and simple segmentation step. We then introduce geometrical and textural features that take into account not only the candidate polyp region, but the surrounding area at multiple scales as well. This way, our proposed CAD algorithm is able to accurately detect candidate polyps by measuring local variations of these features. Candidate patches are then classified using SVM. The whole algorithm is completely automatic and produces state-of-the-art results, achieving 100% sensitivity for polyps greater than $6mm$ in size with less than one false positive per case, and 100% sensitivity for polyps greater than $3mm$ in size with 2.2 false positives per case.

Contents

1 Introduction **1**
- 1.1 The importance of being screened . . . 1
- 1.2 Patient preparation and examination procedure . . . 4
- 1.3 Document Organization . . . 5
- 1.4 Overview of the Proposed System . . . 6
- 1.5 Virtual Colonoscopy CAD Review . . . 7

2 Geometric curve and surface evolution **9**
- 2.1 Differential Geometry of Surfaces . . . 9
  - 2.1.1 Classical definitions . . . 9
  - 2.1.2 Yet another curvature measure (Koenderink) . . . 12
- 2.2 Image processing and surface motions . . . 13
  - 2.2.1 Images and Level Sets . . . 14
  - 2.2.2 Image Diffusion and Curvature Surface Motions . . . 18
  - 2.2.3 Variational formulation - Energy-based snakes . . . 23

3 Colon Segmentation **27**
- 3.1 A challenging problem . . . 27
- 3.2 Classifying CT regions . . . 28
- 3.3 Smoothing . . . 32
- 3.4 Colon surface extraction . . . 37
  - 3.4.1 Thresholding **P** - Marching Cubes . . . 37
  - 3.4.2 Surface evolution techniques - Geodesic Active Regions . . . 39
- 3.5 Summary . . . 40

4 Classification **41**
- 4.1 Candidate detection and feature computation . . . 43
  - 4.1.1 Candidate detection and delineation . . . 43
  - 4.1.2 Intuition in context information . . . 44
  - 4.1.3 Geometrical features . . . 45
  - 4.1.4 Texture features . . . 47
  - 4.1.5 Feature Selection . . . 50
- 4.2 The Class Imbalance problem . . . 53
  - 4.2.1 Meta Cost . . . 54

4.2.2 Cost Sensitive Learner 55
4.2.3 SMOTE 55
4.3 Classifiers 56
4.3.1 Support vector machines (SVM) 56
4.3.2 Classification Trees + AdaBoost 60
4.3.3 Other tested classifiers 64

5 Results and Conclusions 65
5.1 Classification framework and details 65
5.1.1 Database used 65
5.1.2 Implementation details 66
5.1.3 Evaluation scheme and parameters selection 66
5.2 Numerical results 67
5.3 Discussion 70
5.3.1 Small polyps, big polyps, flat lesions 70
5.3.2 Geometric and texture importance 70
5.3.3 Qualitative analysis of false positives 73
5.4 Future work 73
5.5 Conclusions 75

Bibliography 82

Chapter 1

Introduction

1.1 The importance of being screened

Colorectal cancer is the second leading cause of cancer-related deaths in the United States (only surpassed by lung cancer), and the third cause worldwide [75]. The early detection of polyps is fundamental, allowing to reduce mortality rates up to 90% [2]. Nowadays, optical colonoscopy (OC) is the most used detection method due in part to its relative high performance. Optical Colonoscopy is the examination of the colon, usually with a fiber optic camera attached to a flexible tube (endoscope). The preparation process for this examination is very similar to the preparation for Virtual Colonoscopy described in this chapter. After the sedation and that the pain reliever is given, the endoscope is introduced through the anus, and the doctor carefully examines the colon through the video images. A typical image of an optical colonoscopy video is shown in Figure 1.1. OC has the ability to perform therapeutic interventions during the examination. However, this technique is invasive, very expensive, and still prone to miss polyps (in particular small or flat ones), making it hard to use in large screening campaigns.

Computer Tomographic Colonography (CTC), or Virtual Colonoscopy (VC), is a promising alternative technique that emerged in the 90's [71]. It uses volumetric Computed Tomographic data of the cleansed and air-distended colon, and the examination is made by a specialist from the images in a computer. The most common approach is to reconstruct the colon wall and visualize the $3D$ surface with a "fly through" (visualizing the inside of the colon as if performing an optical colonoscopy), see Figure 1.2.

This technique is less invasive and less expensive than optical colonoscopy, and as a consequence, much more suitable for screening campaigns once its performance is demonstrated. CTC also has the potential to outperform OC, in particular for small or flat polyps, or those polyps in regions of the colon where OC has been shown to perform poorly. In addition, regarding optical colonoscopy, only around 70% of the colon is explored visually. Incomplete studies due to obstructing colorectal lesions, colon twists, or anatomical variations are not rare (5% to 15% [6]), and there is the additional risk of perforating the colon. In a large study by Kim *et al.* [35], where about 3000 patients went through OC and another 3000 through VC, seven perforations occurred in the OC while

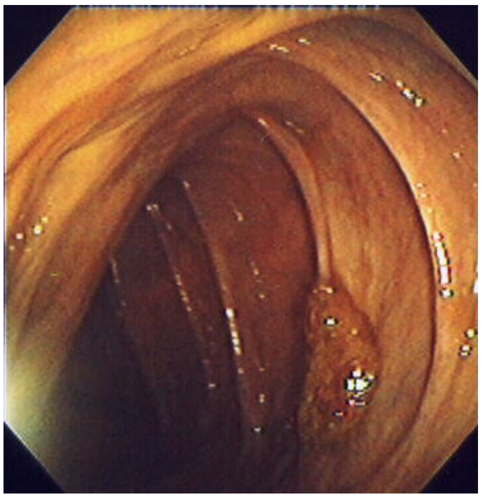

Figure 1.1: Screenshot of typical optical colonoscopy video.

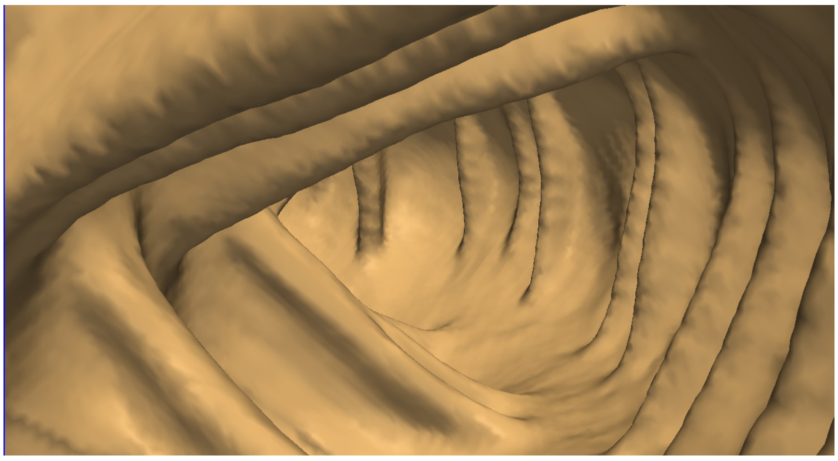

Figure 1.2: Screenshot of typical virtual colonoscopy image.

none were recorded in the VC. The image acquisition process is short, and no recovery period is required, so the patient is able to continue with normal life immediately.

Nevertheless, it takes more than 20 minutes for a trained radiologist to complete a CTC study, and the performance of the overall optical colonoscopy is still considered better. In this regard, polyp Computer-Aided Detection (CAD) algorithms can play a key role, assisting the expert to both reduce the procedure time and improve its accuracy. The aim of these CAD algorithms is not to substitute the radiologists but to help them to achieve higher sensitivities. In this regard, there are three ways to interact with a CAD technique. In the first one, called “first reader”, the expert reviews the results of the flagging algorithm, but does not examine the entire colon. The second one is referred as “concurrent reader”, and here the doctor is able to see the flags during the entire examination. In the third one, called “second reader”, the radiologist makes the whole examination blind to the results of the CAD algorithm, and then reviews the flags produced by the CAD method. Each one of these options has advantages and disadvantages in terms of time or sensitivity. In any case, the false positives produced by the CAD algorithms tend to be easily discarded by the expert [67].

Colon lesions can be classified according to their size, measured in diameter, and according to their morphology, into pedunculated polyps (attached to the colon wall by a stalk), sessile polyps (grow directly from the wall), or flat lesions. See Figures 1.3 and 1.4. Flat lesions are of special interest because they are an important source of false negatives in CTC, and they are around 10 times more likely to contain high-grade epithelial dysplasia[1] [24, 23, 74].

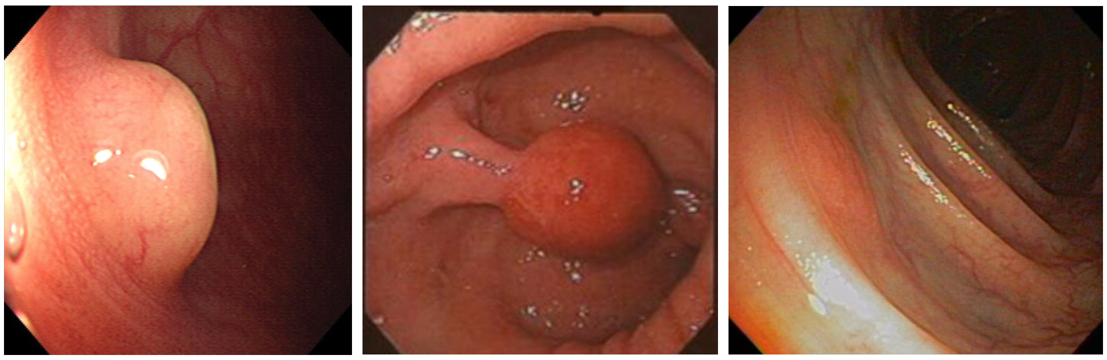

Figure 1.3: Examples of colon polyps as seen on OC: sessile, pedunculated and flat.

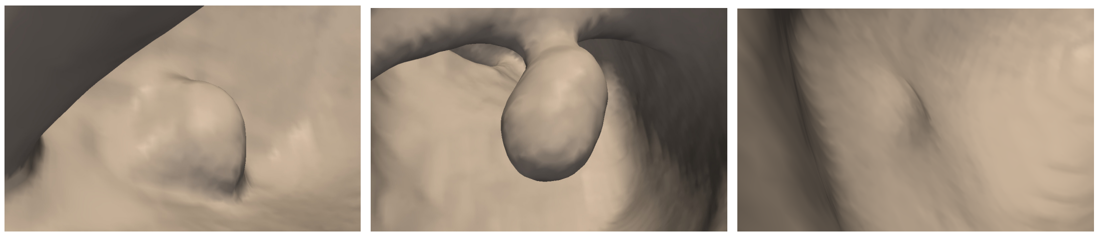

Figure 1.4: Examples of colon polyps as seen on VC: sessile, pedunculated and flat.

There are numerous discussions regarding the potential risks of the polyps according to their size. Even though some authors consider that “small” polyps may not represent risk, the majority of gastroenterologists disagree [3]. Summers claims that one of the major challenges in the field is in increasing sensitivity for smaller polyps [63]. At the same time, Bond [8] declares that the major disadvantage of VC is its current low performance for flat lesions.

[1]An abnormality of development in cells that may become *cancer in situ* or *invasive cancer*.

The goal of the work here presented is to exploit VC precisely to automatically flag colon regions with high probability of being polyps, with special attention to results in challenging small and flat polyps. Toward this aim, we propose a complete pipeline that starts with a novel and simple segmentation step. We then introduce geometrical and textural features that take into account not only the candidate polyp region, but the surrounding area at multiple scales as well. This way, our proposed CAD algorithm is able to accurately detect candidate polyps by measuring local variations of these features. The whole algorithm is completely automatic and produces state-of-the-art results. Preliminary classification results were published in [25]. A paper with the complete work is available in [26].

1.2 Patient preparation and examination procedure

In order to obtain high quality CT images, the colon should be well-cleansed and well-distended. The cleansing is very similar to the preparation for optical colonoscopy. The patient is asked to maintain a clear liquid diet starting 24hs or 48hs before the study and to ingest a cathartic or laxative the night before the examination. This is the process that patients find more unpleasant.

The colon distention is carried out by placing a thin tube into the rectum, and performing the colon insufflation with room air.

A well-cleansed and well-distended colon is ideal for the quality of the study, but it is uncomfortable for the patient, as commented above. A trade-off solution is to combine a laxative or cathartic with *fecal tagging*, which is a method to tag fecal residues by means of a contrast agent, orally administered. Figure 1.5 shows a physically-cleansed and well-distended colon and a fecal tagging CTC image. Due to the gravitational effect, tagged fluid appears as a liquid pool at the bottom of the colon, with a flat horizontal interface with the air region. These latter are the kind of images we worked with, so this presence of liquid is object of discussion in Chapter 3.

Depending on the patient position, some sections of the colon might appear more or less distended, and some sections might have residual material. Therefore, the patient is scanned in two positions, supine and prone, in order to get a good representation of the whole colon. Each scan takes about 20 seconds (depending on the used CT equipment), and during the scan the patient is asked to hold the breath.

The output images are then two sets, each one of them composed by slices of 512×512 pixels. Usually the separation between the slices is around $1mm$, so each CT volume is about $512 \times 512 \times 400$ voxels.

Different tissues have different radiodensity (the property of relative transparency to the passage of x-rays). The intensity of the voxels in the CT image is proportional to the attenuation of the tissue on the Hounsfield scale ($+3071$ to -1024, where 0 corresponds to water and -1000 corresponds to air[2]).

We used a publicly available database of studies described in the last chapter. During

[2]In some cases, the scale is shifted by 1000.

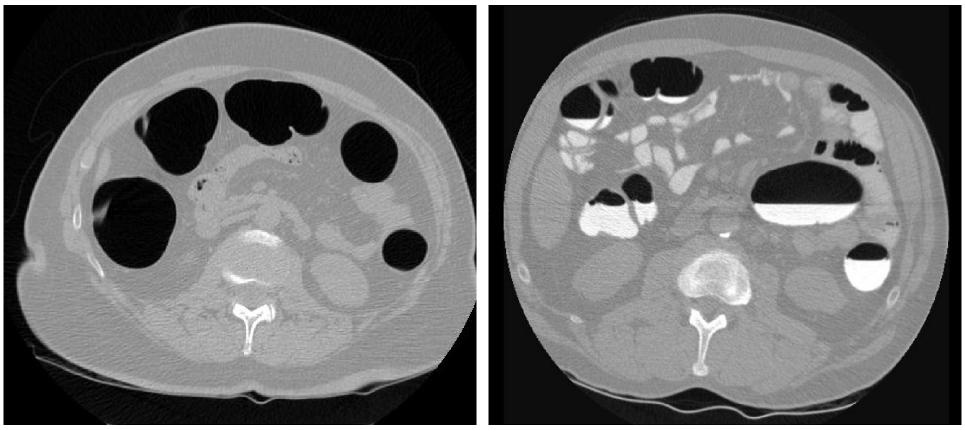

Figure 1.5: Different patient preparations. Left: a physically-cleansed and well-distended colon. Right: a fecal tagging CTC image.

the first months, while working in the feature extraction and classification part, we used the segmentation gently provided by Sergio Aguirre from Echopixel, which we refer to as the "previously provided segmentation". We then developed our own segmentation algorithm.

1.3 Document Organization

The rest of this manuscript is organized as follows.

Finishing this chapter, in Section 1.4 we describe the pipeline of the proposed algorithm for polyp Computer-Aided Detection and in Section 1.5 we briefly review prior related work.

In Chapter 2 we lay the theoretical foundations for the rest of the thesis. We first recall some differential geometry definitions and results, with special focus on shapes and curvatures of surfaces. Then, we introduce some classical image processing ideas and techniques, including the representation of an image through its level sets, some basic Partial Differential Equations problems, the Level Set Method [48], motion of surfaces by curvatures and active-contour segmentation techniques.

In Chapter 3 we tackle the colon segmentation problem. The chapter starts with an introduction to the difficulties this problem presents, and the desirable properties of the results. We present a preprocessing stage to deal with the air-liquid composition of the inside of the colon, and then the smoothing step, with special attention to the deformation of the polyp-like structures. The chapter ends with the colon surface extraction from the processed and smoothed volume.

Chapter 4 is devoted to the classification. The multi-scale candidate detection stage is described first, followed by the (geometrical and texture) feature extraction. These features take into account the local information around each candidate region, leading to a higher classification performance. The class imbalance problem is presented with some approaches to deal with it. Finally, we describe the tested classifiers (SVM, classifier trees).

Chapter 5 is the last one; it includes a description of the used database, the evaluation scheme and the numerical results, which we compare with the state-of-the-art results. Then, a discussion is given, and the chapter ends with some lines of future work and conclusions.

1.4 Overview of the Proposed System

The main goal we are addressing in this work is to highlight/flag all the candidate polyps, so the radiologist can quickly check them. It is crucial to minimize the false negatives (missed polyps), keeping a reasonable false positives number. We achieve this by a four-steps process, Figure 1.6, which is completely automatic and constitutes the entire end-to-end algorithm, from data to candidate polyps flagging.

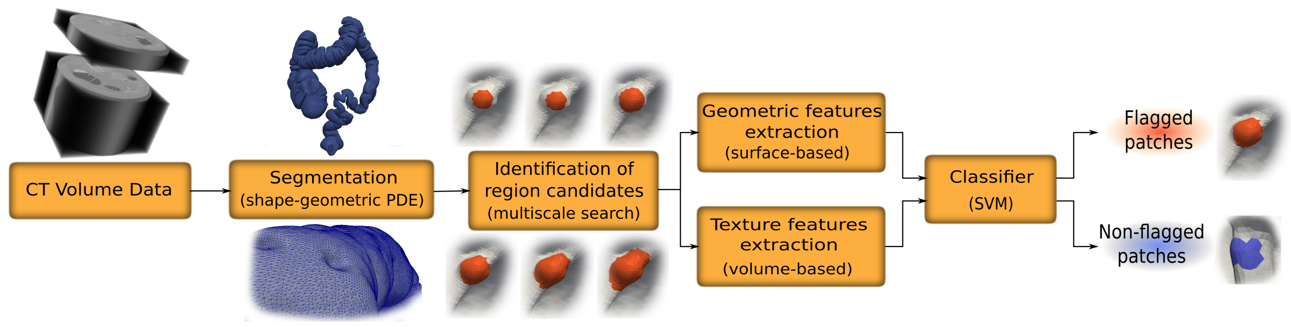

Figure 1.6: Basic pipeline of the proposed polyp flagging system.

The input of the system is, of course, the CT volume data. In the first step, the colon segmentation, this CT data is processed in order to extract a surface that corresponds to the colon wall. One of the main difficulties in this stage is the presence of liquid, so first we classify the voxels as “exterior” and “interior” voxels of the colon. Then we use this classification (and a smoothing process) to extract the surface that represents the colon wall. The output is a $3D$ mesh of this surface.

This mesh contains about a million points, so we have to determine those regions that are candidates of being polyps. This is done in the second step, where from the segmented mesh we perform a multi-scale search of candidates in order to capture the appropriate polyp size. The result is a set of candidate patches, which we have to classify (or flag) as polyps or non-polyps.

The CT volume data, the segmented colon surface, and the candidate patches are the input to the third step, where we compute geometrical and texture features. The geometrical features, computed directly from the surface, are strongly based on curvatures and other measures of the local shape. The texture features, computed from the CT volume, quantify the homogeneity properties of the region, in terms of the gray level.

Both the geometrical and texture features take into account the information around the candidate region, with the spirit of looking for differences between the candidate patch and the normal tissue around it.

The final step consists of a machine learning algorithm that classifies polyps and non-polyps patches from the computed features.

In the following chapters we describe each of these steps in detail.

1.5 Virtual Colonoscopy CAD Review

Automatic polyp detection is a very challenging problem, not only because the polyps can have different shapes and sizes, but also because they can be located in very different surroundings. Most of the previous work on CAD of colonic polyps is based on geometric features, some using additional CT image density information, but none of them takes into account the (geometric and texture) information of the tissues *surrounding* the polyp. This is a crucial issue since it is well known that the tissue properties of the colon may vary with location. This local and adaptive differential analysis is part of the contributions of this work.

Early work on CAD methods by Vining *et al.* [70], is based on the detection of abnormal wall thickness. Summers *et al.* [64], detect polyps greater than $10mm$ by computing mean curvatures, average principal curvatures and sphericity ratio, and then classifying with a committee of SVMs. They present results over a large screening patient population, but the amount of polyps is comparable with the rest of the works. Yoshida *et al.* [76], use the *shape index* (defined later in this thesis) and *curvedness* as geometric features, applying fuzzy clustering and then using directional gradient concentration to reduce false positives. Paik *et al.* [49], also use geometrical features, computing the Surface Normal Overlap (SNO) instead of calculating curvatures. Wang *et al.* [73], compute a global curvature, extract an ellipsoid, and analyze morphological and texture features on this ellipsoid. They reach 100% sensitivity with a relative low false positives (FP) rate, using heuristic thresholds and texture features. Hong *et al.* [32], map the 3D surface to a rectangle, use 2D clustering, and reduce false positives with shape and texture features. Sundaram *et al.* [65], compute curvatures via the Smoothed Shape Operators method, and use principal curvatures and Gaussian curvatures to detect polyps. Götkürk *et al.* [29], propose a technique to reduce the false positives based on features calculated from three random orthogonal sections, and then classifying with SVM, but as it is presented as a FP reduction scheme, no false positive per patient result is reported. All these described techniques based on local geometric computations suffer from a high dependence on the regularity of the polyp shape itself, ignoring how pronounced it is with respect to the surrounding area. Using geometry alone is also very sensitive to small differences in the colon segmentation, always prone to be present.

More recently, van Wijk *et al.* [69] proposed a PDE motion that flattens mainly the polyp-like shapes, and then they consider the difference between the original and the processed images. The main drawback of this approach is that the resulting flattened polyps look like a flat lesion, therefore the algorithm does not detect flat lesions. Suzuki *et al.* [66] use artificial neural networks to reduce the false positive number of the previous algorithm by Yoshida *et al.* described above [76]. Although there were no polyps submerged in fluid in the database, the results are very promising, achieving 96.4% sensitivity (over 28 polyps) with 1.1 FP per case. However, the sensitivity is not perfect, an evaluation with small lesions and polyps submerged in liquid should be performed, and the FP number can be further improved. Konukoglu *et al.* [38] introduced a preprocessing stage that enhances the polyps via a PDE evolution based on the heat equation, and showed how it improves the CAD performance. Proprietary algorithms [7, 67], have been reported as well, but with no better results than the methods mentioned above. Although the comparison of the experimental results is delicate since different databases were used, all these approaches can be improved as here demonstrated, either detecting a more general class of lesions or directly on the classification performance.

Technique	# Polyps	Size	Sensit.	FP p/study	Validation
Summers *et al.* [64]	≈ 30	$> 10mm$	89.3%	2.1	T+T
Summers *et al.* [64]	≈ 120	$> 6mm$	60%	8	
Yoshida *et al.* [76]	12	$> 5mm$	100%	2	LOO
Paik *et al.* [49]	7	$> 10mm$	100%	7	LOO
Wang *et al.* [73]	61	$> 4mm$	100%	2.68	LOPO
Hong *et al.* [32]	≈ 120	$> 5mm$	100%	3	T+T
Sundaram *et al.* [65]	20	$> 10mm$	100%	18	Test (Unsup)
Sundaram *et al.* [65]	122	$> 2mm$	80%	24	
van Wijk *et al.* [69]	57	$> 6mm$	95%	5	LOPO
van Wijk *et al.* [69]	32	$> 10mm$	95%	4	
Suzuki *et al.* [66]	28	$> 5mm$	100%	1.1	T+T
Bogoni *et al.* [7]	21	$> 5mm$	90%	> 3	LOPO and T+T
Taylor *et al.* [67]	32	$> 6mm$	81%	13	
Nappi it et al. [45]	12	$> 5mm$	100%	2.4	LOPO

Table 1.1: Numerical comparison of the reviewed methods.

The results reported by the algorithms presented above correspond to databases containing polyps greater than $6mm$ in size (or greater than $10mm$ in some databases). Table 1.1 summarizes the numerical results of the presented algorithms, adding results for smaller polyps when reported by the authors. The validation method is also listed, where LOO, LOPO and T+T stands for Leave-one-out, Leave-one-patient-out and disjoint Training/Testing datasets respectively. The method by Sundaram *et al.* is unsupervised so the whole dataset is used for testing. Note that the majority of the reviewed works use a cross-validation technique. To the best of our knowledge, no algorithm reported in the literature can detect small polyps properly. On the other hand, for polyps larger than $6mm$ in size, no algorithm can achieve 100% sensitivity with less than one false positive per study. Therefore, it is important to keep improving these techniques.

Chapter 2

Geometric curve and surface evolution

Since the main idea is to reconstruct the colon surface from the CT images and detect polyps based on some geometric features among others, several notions of differential geometry and basic image processing are used both in the segmentation and feature extraction stages. In this chapter we present the definitions and results that support the algorithms in the following chapters.

First we introduce the definitions of surface, its classical curvatures and the curvature measures used along this work. Then we discuss the representation of an image through its level sets and we study some evolutions governed by Partial Differential Equations (PDEs). The chapter ends with a very brief description of some segmentation techniques based on energy minimization and solved using PDEs.

2.1 Differential Geometry of Surfaces

2.1.1 Classical definitions

Most of the concepts and computations handled in this thesis need a basic understanding of differential geometry, in particular of the notion of surfaces curvatures. In this section we briefly recall these concepts, which were mostly developed and formalized by Gauss in his 1827 work *Disquisitiones generales circa superficies curvas*[1]. An excellent and very enjoyable treatment of the subject can be found in the classic book by Do Carmo [19], or in the second one of the great volumes by Spivak [62].

The object of study in this section are regular surfaces. These are, in a way, two-dimensional subsets of R^3: they can be covered with pieces of a plane by deforming and arranging them. Locally, a surface can be well approximated by a plane.

Definition 2.1.1. *(Regular Surface) A regular surface is a subset $\mathcal{S} \subset \mathbb{R}^3$ such that for each point $p \in S$ there exist an open set $U \subset R^2$, a neighborhood $V \subset R^3$ and*

[1]A translation of this work can be found in [62].

a homeomorphism $\varphi : U \to V \cap \mathcal{S}$ *differentiable, and for each* $q \in U$ *the differential* $d\varphi_q : \mathbb{R}^2 \to \mathbb{R}^3$ *is one-to-one.*

These conditions guarantee the existence of a tangent plane at each point, and avoid self-intersections, which allows to speak about *the* tangent plane at a point. The map $\varphi(u, v)$ is called a *parameterization* and its partial derivatives are denoted by φ_u and φ_v.

The tangent plane to $\mathcal{S}$ at p, denoted by $T_p(\mathcal{S})$, is the vector subspace $dh_p(\mathbb{R}^2)$, it is the plane that best fits the surface at p. Alternatively, the tangent plane $T_p(\mathcal{S})$ can be characterizated as the set of the tangent vectors $\alpha'(0)$ of differentiable parameterized curves $\alpha : (-\varepsilon, \varepsilon) \to \mathcal{S}$ such that $\alpha(0) = p$. The proof that these definitions are equivalent is straightforward.

As each tangent plane is a subspace of R^3, the inner product of R^3 induces an inner product on each tangent plane $T_p(\mathcal{S})$. This leads to the definition of the *First fundamental form* $I_p : T_p(\mathcal{S}) \to \mathbb{R}$ as the inner product of the vector $\vec{w}$ with itself: $I_p(\vec{w}) = \langle \vec{w}, \vec{w} \rangle = |\vec{w}|^2$.

Although the first fundamental form is one of the most important concepts and it allows us to compute length of curves, angles between tangent vectors or areas of regions, we will focus on the second order derivatives, which shed some light on the notion of *shape*. This concept of shape given in this chapter is a characterization in terms of the curvatures and it is profusely used during the thesis.

Given a parameterization $\varphi : U \to V \cap \mathcal{S}$ around p with the corresponding tangent plane $T_p(\mathcal{S})$, an orientation of the plane $T_p(\mathcal{S})$ (counter-clockwise for example) induces a local orientation of the surface near the tangent plane, $\varphi(U)$ (see Figure 2.1). We say that a surface is orientable if these local orientations can be made in such a way that at any intersection of neighborhoods the corresponding orientations coincide. Formally, that is: given two parameterizations $\varphi_1 : U_1 \to \mathcal{S}$ and $\varphi_2 : U_2 \to \mathcal{S}$ such that $W = \varphi_1(U_1) \cap \varphi_2(U_2) \neq \phi$, let $f = \varphi_2^{-1} \circ \varphi_1|_{\varphi^{-1}(W)}$, then $det(d_q f) > 0 \quad \forall q \in \varphi_1^{-1}(W)$.

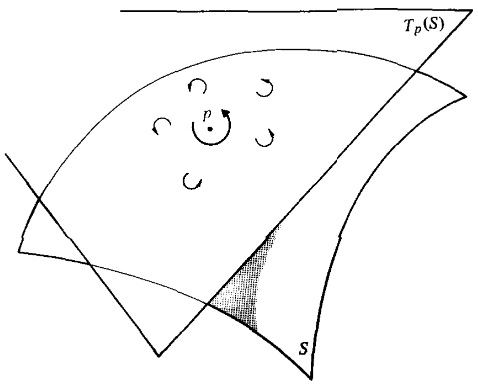

Figure 2.1: Surface with its tangent plane at p, and the orientation induced by $T_p(\mathcal{S})$.

An alternative definition can be given by means of the surface normals. Given a parameterization φ, we can choose a unit normal vector at each point q by

$$\vec{\mathcal{N}} = \frac{\varphi_u(q) \wedge \varphi_v(q)}{||\varphi_u(q) \wedge \varphi_v(q)||}$$

if the surface is orientable, then it admits a differentiable field of unit normals defined on the whole surface. This map $\mathbf{N} : \mathcal{S} \to S^2$, that assigns to each point q of the surface its corresponding normal vector $\vec{\mathcal{N}}(q)$, is the so called *Gauss map*. As the map $\mathbf{N}$ goes from $\mathcal{S}$ to S^2, its differential at p goes from $T_p(\mathcal{S})$ to $T_{\mathbf{N}(p)}(S^2)$, and moreover, since the planes $T_p(\mathcal{S})$ and $T_{\mathbf{N}(p)}(S^2)$ are parallel (both are perpendicular to the normal $\vec{\mathcal{N}}$ at p), $d_p\mathbf{N}$ can be seen as a linear map on $T_p(\mathcal{S})$ (an endomorphism $d_p\mathbf{N} : T_p(\mathcal{S}) \to T_p(\mathcal{S})$). A closer look at $d_p\mathbf{N}$ gives some intuition about its relation with the curvatures, detailed below. Let $\mathcal{C}$ be a regular curve in $\mathcal{S}$ with $\mathcal{C}(0) = p$, and consider the curve $\mathbf{N} \circ \mathcal{C}(t) = \alpha(t)$, this is the path drawn over S^2 of the normals to $\mathcal{S}$ as we move along $\mathcal{C}(t)$. The vector $\alpha'(0) = d_p\mathbf{N}(\mathcal{C}'(0))$ is a vector in $T_p(\mathcal{S})$ that measures the rate of change of the normal vector $\vec{\mathcal{N}}$ restricted to $\mathcal{C}$ at $t = 0$. Therefore, $d_p(\mathbf{N})$ measures how fast the normal $\vec{\mathcal{N}}$ gets away from $\vec{\mathcal{N}}(p)$, in the considered direction. For curves (where there is only one direction) this is the curvature, in surfaces this gives the linear map $d_p\mathbf{N}$.

The map $d_p\mathbf{N}$ results self-adjoint, this allows us to define a quadratic form $II_p : T_p(\mathcal{S}) \to \mathbb{R}$ called the *Second fundamental form* by $II_p(\vec{v}) = -\langle d_p\mathbf{N}(\vec{v}), \vec{v}\rangle$, which encompass the surface curvature information.

We will now define the curvatures of $\mathcal{S}$ at each point and link them with the second fundamental form and the map $d_p\mathbf{N}$. Let us consider a point $p \in \mathcal{S}$ and a regular curve $\mathcal{C}$ passing through it with normal $\vec{n}$, let κ be the curvature of $\mathcal{C}$ at p and $\cos\theta = \langle \vec{n}, \vec{\mathcal{N}}\rangle$. The number $\kappa_n = \kappa \cos\theta$ is called the *normal curvature* of $\mathcal{C}$ at p.

Observe that κ_n is the projection of $\kappa\vec{n}$ over $\vec{N}$ (see Figure 2.2). The sign of the normal curvature then depends on the chosen orientation of $\mathcal{S}$.

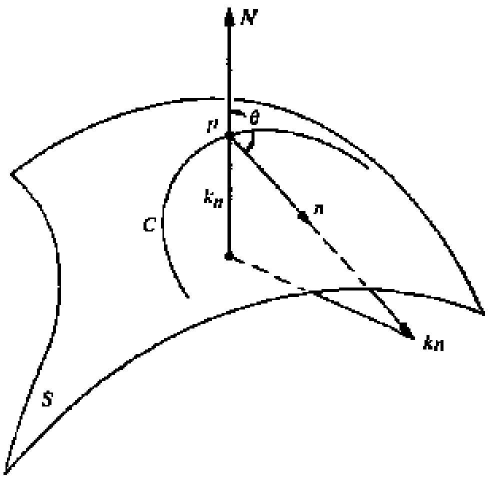

Figure 2.2: Surface $\mathcal{S}$ and the normal curvature for the curve $\mathcal{C}$.

Returning to II_p, if $\mathcal{C}$ is a regular curve parameterized by arc-length such that $\mathcal{C}(0) = p$ and $\mathcal{C}'(0) = \vec{v}$, then its normal curvature verifies $\kappa_n = II_p(\vec{v})$.

Therefore, κ_n does not depend on the particular curve $\mathcal{C}$: all curves with the same tangent line at p have the same normal curvature. With this result, we can speak about *the* normal curvature at p along a certain direction and forget about the auxiliary curves $\mathcal{C}$.

Now that we have a normal curvature for each direction on $T_p(\mathcal{S})$ we can concern about the maximal and minimal values of κ_n over all directions. These values, κ_1 and κ_2, the maximal and minimal curvatures respectively, are called the *principal curvatures.* The corresponding *principal directions*, say e_1 and e_2, are orthogonal, and if we take another direction making an angle θ with e_1, then its normal curvature is $\kappa_n = \kappa_1 \cos^2\theta + \kappa_2 \sin^2\theta$. Although this definition and the relationship between κ_n, κ_1 and κ_2 are very illustrative[2], it is not clear how to compute the principal curvatures in an efficient way.

Since $d_p\mathbf{N}$, the differential of the Gauss map, is self-adjoint, by virtue of the spectral theorem it admits an orthonormal basis of eigenvectors e_1, e_2. These happen to be the principal directions and, of course, its corresponding eigenvalues are the principal curvatures κ_1, κ_2. Since the determinant and the trace of a linear map do not depend on the chosen basis, we define then the *Gaussian curvature* K at p as the determinant of $d_p\mathbf{N}$ and the *Mean curvature* $\mathcal{H}$ at p as the negative half trace of $d_p\mathbf{N}$. More precisely:

$$K = \kappa_1\kappa_2$$

$$\mathcal{H} = \frac{\kappa_1 + \kappa_2}{2}$$

The last classic concept in this section is the concept of geodesic, which is related to how the surface geometry changes the way of measuring distances. Given two points $p, q \in \mathcal{S}$, a path α over $\mathcal{S}$ of minimal length is called a *geodesic curve* and the length of that curve is called the *geodesic distance* between p and q. On the plane, the shortest path is the straight line, but in a surface the geodesic curve may be not unique.

2.1.2 Yet another curvature measure (Koenderink)

We have presented the principal curvatures κ_1, κ_2, and the Mean and Gaussian curvature $\mathcal{H}, K$, now we will add to these classical measures a couple more that capture some structure, that will be helpful to describe the polyps' geometry.

In order to characterize the polyp-like shapes, it is obvious that the measure should not depend on the orientation nor the position, but it is also quite intuitive that it should be invariant to homotheties, so it capture the shape itself. The curvatures presented until now vary with the scale, for example two spheres with different radii have different curvatures (principal, mean or Gaussian), so actually they do not show that the spheres have essentially the same shape.

The descriptors defined next, have the property of decoupling the "shape" from the "size" (or "how pronounced the shape is"). These are the so-called *Shape Index* and

[2]Euler was the first to work with these relations.

Curvedness, defined in terms of the principal curvatures as follows [37]:

$$SI := -\frac{2}{\pi}\arctan\left(\frac{\kappa_1+\kappa_2}{\kappa_1-\kappa_2}\right),$$

$$R := \sqrt{\frac{\kappa_1^2+\kappa_2^2}{2}}\ .$$

The factor $-\frac{2}{\pi}$ in SI is to map all shapes to the $[-1, 1]$ interval, and the $\frac{1}{2}$ factor in R is to normalize the scale such that the surface of the unit sphere has unit curvedness.[3] Under this coordinate transformation, the (κ_1, κ_2) plane is transformed into the (SI, R) plane, in a sort of a polar coordinate transformation. While the value of SI is scale-invariant and measures the local shape of the surface, the value of R indicates how pronounced it is. Figure 2.3 shows different shapes and their corresponding Shape Index, for example the saddle point (opposite principal curvatures) has $SI = 0$, the rut has $SI = 0.5$, the ridge has $SI = -0.5$ and the spherical cup and cap have $SI = 1$ and $SI = -1$ respectively. Opposite values of SI correspond to the same shape with the opposite normal direction. As we will see in Section 4.1.3, due to the chosen orientation for our surface, Shape Index values close to -1 are of special interest for polyp detection.

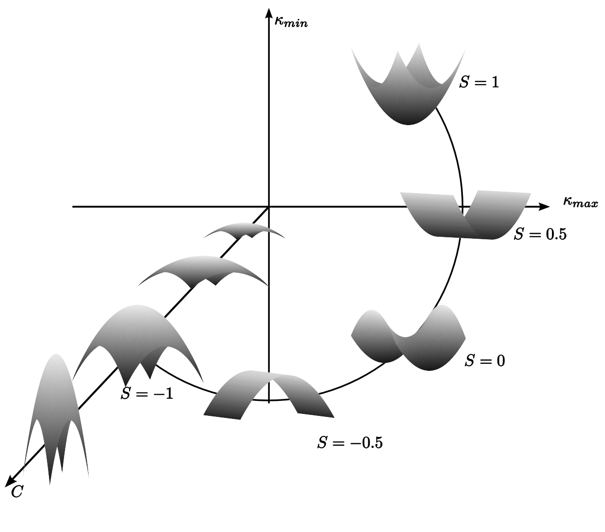

Figure 2.3: Some shapes and their corresponding *Shape Index* values.

2.2 Image processing and surface motions

The topic list in this section is classic and well known, but it is still very delightful at the same time, because of the elegance of the methods and results. A very concise summary

[3]Sometimes, an alternative measure $C := \frac{2}{\pi}\ln(R)$ is used instead of R.

is given in what follows. Although the focus is on the concrete tools applied along this work, in some cases we point out some side comments and relations with other methods.

The tools presented here are the basis for various tasks of the Chapter 3. First we introduce the level set representation of curves and surfaces and the Level Set Method introduced by Sethian and Osher [48]. Then some particular concepts of geometric curve and surface evolution are discussed with the goal of image smoothing. Finally a very brief introduction to segmentation through active contours is given.

All these topics are covered in a great variety of books with a wide range of approaches and scopes. We highlight two with an overview of the field: [54, 44].

2.2.1 Images and Level Sets

Let us interpret an image as a real function $u : U \to \mathbb{R}$, with domain $U \subset \mathbb{R}^2$ or $U \subset \mathbb{R}^3$. We will use a bold letter to name vectors, for example $\mathbf{x} = (x, y)$ or $\mathbf{x} = (x, y, z)$.

A very common representation of an image is through its lower level sets, defined as

$$\mathcal{X}_\lambda = \{\mathbf{x} : u(\mathbf{x}) \geq \lambda\}.$$

This representation has the property of being invariant to contrast changes[4], and it contains all the information of the image, in fact, the image can be reconstructed from its level sets by

$$u(\mathbf{x}) = \sup\{\lambda : \mathbf{x} \in \mathcal{X}_\lambda\}.$$

Another description of an image, related to the level sets, is in terms of level lines (or level surfaces in the $3D$ case) which are the topological boundaries of the connected components of the level sets. As we deal with CT volumes in this thesis, we will work directly with the $3D$ case, first recalling the Implicit Function Theorem.

Theorem 2.2.1. *Let $u : \mathbb{R}^3 \to \mathbb{R}$ be a C^1 function in a neighborhood of $\mathbf{p_0} = (x_0, y_0, z_0)$ such that $\nabla u(\mathbf{p_0}) \neq 0$, then there exists an open set $U \subset \mathbb{R}^2$, an open set $V \subset \mathbb{R}$ and a C^1 function $\varphi : U \to V$ such that*

$$u(\mathbf{p}) = 0 \Leftrightarrow z = u(x, y).$$

In other words, the set $\{\mathbf{p}, u(\mathbf{p}) = 0\}$ can be locally described as the graph of the function φ. We obtain a surface which we will call "level surface of u around $\mathbf{p}$". Therefore, given a number $\lambda \in \mathbb{R}$ we can decompose $u^{-1}(\lambda)$ into a finite set of embedded manifolds.

Luckily, if the function u is smooth enough, Sard's theorem states that for almost every $\lambda \in \mathbb{R}$ the set $u^{-1}(\lambda)$ is nonsingular, which means that $\nabla u(\mathbf{p}) \neq 0\,, \forall \mathbf{p} \in u^{-1}(\lambda)$.

More precisely, Sard's theorem states the following.

[4]that is, if u is an image and $v = g(u)$ a contrast changed version (g continuous and increasing), then there is a 1-to-1 correspondence between the level sets of u and v, as it is obvious.

Theorem 2.2.2. *(Sard) If $f : \mathbb{R}^n \to \mathbb{R}^m \in C^k$ with $k \geq \max(1, n-m+1)$, and $\mathcal{A}$ is the set of the points $\mathbf{x}$ at which the Jacobian matrix of f has rank $r < m$ (i.e. the set of critical points), then $f(\mathcal{A}) \subset \mathbb{R}^m$ has zero Lebesgue measure.*

We will make use of this representation from two slightly different viewpoints. On the one hand, we will process an image in such a way that its level surfaces behave according with some desirable properties. This is the viewpoint used in the smoothing section (2.2.2). On the other hand, we will be interested in the motion of surfaces themselves for segmentation, as discussed in the last section of this chapter, so we will define an auxiliary function such that the surface of interest is a level set of the auxiliary function.

In what follows we focus in evolving surfaces, meaning a family of surfaces $\mathcal{S}_t$ where t parameterizes the "time"; these families obey certain Partial Differential Equations (PDEs). Considering the flow $\mathcal{S}_t$ as level surfaces of a family of images u_t, it is crucial to relate the equations of the motion of both representations.

For instance, let $\mathcal{S}$ be a surface and suppose that we want to make it evolve according to the following equation:

$$\frac{\partial \mathcal{S}(t)}{\partial t} = \beta \vec{\mathcal{N}}_{\mathcal{S}} \ , \tag{2.1}$$

where $\vec{\mathcal{N}}_{\mathcal{S}}$ is the unit normal to the surface and β is a function computed on $\mathcal{S}$. Let us consider that at each time t the surface $\mathcal{S}_t$ is the zero level set of the function $u : \mathbb{R}^+ \times \mathbb{R}^3 \to \mathbb{R}$ (where the $\mathbb{R}^+$ is the time domain), and a local parameterization of the surface $\varphi : \mathbb{R}^+ \times U \to \mathbb{R}^3$, $U \subset \mathbb{R}^2$, such that $\varphi(t, \cdot)$ is a local parameterization of the surface $\mathcal{S}_t$. Therefore, the following equation is satisfied

$$u\left(t, \varphi(t, \mathbf{p})\right) = 0 \quad \forall \mathbf{p} \in U, t \in \mathbb{R}^+ \ .$$

Assuming regularity, by differentiating this equation with respect to time t, applying the chain rule, we obtain

$$\frac{\partial u}{\partial t} + \left\langle \nabla u, \frac{\partial \varphi}{\partial t} \right\rangle = 0 \ .$$

Here there is an abuse of notation, since ∇u stands for the gradient taken with respect to the spatial variables, leaving aside the temporal variable t.

Recalling that the original motion of the surface was $\frac{\partial \varphi}{\partial t} = \beta \vec{\mathcal{N}}$, we can rewrite the last equation, obtaining

$$\frac{\partial u}{\partial t} + \left\langle \nabla u, \beta \vec{\mathcal{N}} \right\rangle = 0 \ .$$

Finally, as the gradient of u is perpendicular to the level sets, we obtain the unit normal by $\vec{\mathcal{N}} = -\frac{\nabla u}{||\nabla u||}$, where the sign is minus if u is negative inside the surface and positive outside. Consequently,

$$\frac{\partial u}{\partial t} - \left\langle \nabla u, \beta \frac{\nabla u}{||\nabla u||} \right\rangle = 0 \quad \Rightarrow \quad \frac{\partial u}{\partial t} = \beta \frac{1}{||\nabla u||} \left\langle \nabla u, \nabla u \right\rangle$$

Then, the corresponding evolution viewing $\mathcal{S}_t$ as the zero level sets of u_t is

$$\frac{\partial u}{\partial t} = \beta ||\nabla u|| \tag{2.2}$$

This means that when an image u evolves according to (2.2), its level sets evolve according to (2.1).

When the goal is to make evolve a surface (or curve), this last trick of interpreting the surface as a level set of a higher dimensional image and make evolve this image, is known as the Level Set Method (LSM), developed by Sethian and Osher [48].

When dealing directly with the surface it is necessary to parameterize it and to use a Lagrangian representation to compute the evolution. This implies keeping track of the markers and redistributing the markers after the iterations as they can come close together or far away during the evolution and to solve other numerical issues. Additionally, when evolving a surface its topology can change (split or merge), so keeping track of these changes with markers becomes a tedious work. One of the advantages of the LSM is that these problems are automatically handled. The flow given by Equation (2.2) corresponds to an Eulerian representation because it is written in terms of a fixed reference frame, so classical numerical methods can be used to solve the PDE. On the other hand, in the higher dimensional space topology changes are naturally handled by the embedding (see Figure 2.4).

In all the motions described below the term β is a function of the curvatures. Hence, in order to use the Level Set Method, it is necessary to compute the curvatures from the image u in which the surface is embedded.

We end up this section presenting a different expression for the curvatures, whose aim is clear from what was recently seen. We have already mentioned that, if we have the surface $\mathcal{S}$ represented as the zero level set of a function $u : \mathbb{R}^3 \to \mathbb{R}$, i.e. $\mathcal{S} = \{(x, y, z) \in \mathbb{R}^3 : u(x, y, z) = 0)\}$, then the unit normal is

$$\vec{\mathcal{N}} = -\frac{\nabla u}{||\nabla u||}.$$

It can be shown (see for example [60],[68]) that the mean curvature $\mathcal{H}$ and the Gaussian curvature K can be computed as follows:

$$\mathcal{H} = div\left(\frac{\nabla u}{||\nabla u||}\right) = \frac{1}{(u_x^2+u_y^2+u_z^2)^{3/2}} \Big[(u_{yy}+u_{zz})u_x^2 + (u_{xx}+u_{zz})u_y^2 + (u_{xx}+u_{yy})u_z^2 \\ -2u_xu_yu_{xy} - 2u_xu_zu_{xz} - 2u_yu_zu_{yz}\Big] ,$$

$$K = \frac{1}{(u_x^2+u_y^2+u_z^2)^2}(u_x^2(u_{yy}u_{zz}-u_{yz}^2) + u_y^2(u_{xx}u_{zz}-u_{xz}^2) + u_z^2(u_{xx}u_{yy}-u_{xy}^2) \\ +2[u_xu_y(u_{xz}u_{yz}-u_{xy}u_{zz}) + u_yu_z(u_{xy}u_{xz}-u_{yz}u_{xx}) + u_xu_z(u_{xy}u_{yz}-u_{xz}u_{yy})]) .$$

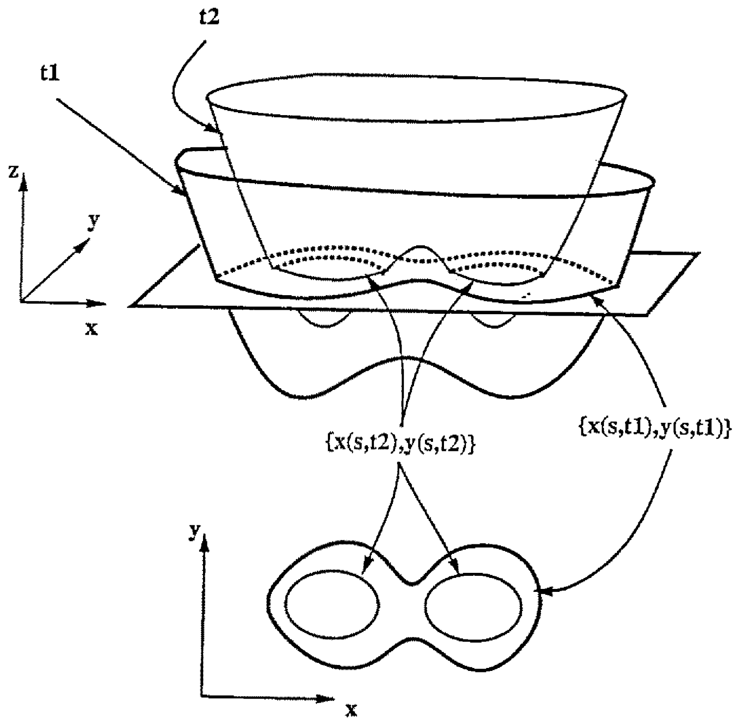

Figure 2.4: Level Set Method: the embedding function at times t_1 and t_2, with their corresponding iso-level sets. Note how topological changes are automatically handled. Figure from [55].

Although this formulae are useful and necessary when solving PDEs with the Level Set Method like the ones presented in the rest of this chapter, in certain cases some numerical problems may appear, due to the fact that these curvatures involve second order derivatives. In particular, the principal curvatures (which are necessary to compute the shape index) are obtained by solving a second order equation, leading to

$$\kappa_{1,2} = \mathcal{H} \pm \sqrt{\mathcal{H}^2 - K}\,,$$

so very little errors in the estimation of $\mathcal{H}$ and K may became important errors in the estimation of the principal curvatures. In order to overcome this difficulty we directly compute the principal curvatures as follows [43]. Given a vector v, let P_v be the operator that represents the projection onto the orthogonal complement of v, that is, $P_v = I - \frac{vv^T}{|v|^2}$, let H_u be the Hessian matrix of the function u and consider the symmetric matrix

$$J = \frac{1}{|\nabla u|} P_{\nabla u} H_u P_{\nabla u}\,. \tag{2.3}$$

The orthogonal complement of ∇u is the tangent plane, where the second fundamental form is defined. It is easy to see that ∇u is a eigenvector of J corresponding to the zero eigenvalue, due to the projection. It can be shown that the other two eigenvalues are the principal curvatures and the corresponding eigenvectors are the principal directions.

This computation scheme leads to much more accurate estimation of the curvatures, and therefore better estimations of the shape index. Both techniques (the direct formulae and this latter one) yield very similar results for the mean curvature $\mathcal{H}$. The important difference is on the computation of the Gaussian curvature K, where the latter technique leads to much more stable results. When the absolute value of the Gaussian curvature K is small, sometimes the different estimations have different signs, leading to very significant errors in the shape index estimation, since SI is very sensitive to the signs.

2.2.2 Image Diffusion and Curvature Surface Motions

We begin this section with the theory for the $2D$ case, and then we extend some results (those that can be extended) to the $3D$ case, which is the one that corresponds to the problem addressed in this work.

Possibly the most classic technique to smooth an image is by filtering it with a Gaussian filter G_t with variance t, which gives the level of smoothing or *scale*. This is, if u_0 is the original image, the smoothed image is obtained by:

$$u(\mathbf{x}, t) = (u_0 * G_t)(\mathbf{x}).$$

The theory of the image representation through these different smoothed versions is called *scale space*.

A simple yet elegant result states that if $u(\mathbf{x}, t)$ is obtained via this Gaussian filtering, then it satisfies the heat equation[5]

$$\frac{\partial u(\mathbf{x}, t)}{\partial t} = \triangle u(\mathbf{x}, t) \qquad u(\mathbf{x}, 0) = u_0(\mathbf{x})$$

which is one of the most studied PDEs. An heuristic proof is straightforward: the Gaussian function satisfies the heat equation, the PDE is linear and the operations (derivatives and convolutions) are linear. The Gaussian function is then a fundamental solution of the heat equation. This statement can be further sharpened, and under certain regularity hypotheses, results about the existence, uniqueness and uniform convergence of the heat equation solutions can be given [44].

The first comment is that the heat equation is not contrast invariant and it is isotropic, that is, the diffusion is the same in all directions, independently on the image, in particular it does not preserve edges. Therefore, we will study alternative motions to improve these aspects.

The set of the level sets constitute a complete representation of an image. Thus, it is important to study how the image level sets are affected by image diffusions.

Now, if we make evolve a curve $\mathcal{C}$ with parameterization $(x(t), y(t))$, according to the mentioned heat equation, that is:

[5]Moreover, iterating the convolution with any symmetric kernel, not necessarily the Gaussian, with variance tending to zero, asymptotically converges to the solution of the heat equation .

$$\frac{\partial x}{\partial t} = \frac{\partial^2 x}{\partial p^2} \quad , \quad \frac{\partial y}{\partial t} = \frac{\partial^2 y}{\partial p^2} \ ,$$

then, although $x(p,t), y(p,t) \in C^\infty$ and this equation can be solved with a numerically stable technique, the smoothness of the coordinates does not imply the smoothness of $\mathcal{C}$, and moreover, the evolution depends on the chosen parameterization, it is not intrinsic.

However, if we use the arc-length parameterization instead, we obtain the set of equations

$$\frac{\partial x}{\partial t} = \frac{\partial^2 x}{\partial s^2} \quad , \quad \frac{\partial y}{\partial t} = \frac{\partial^2 y}{\partial s^2}$$

that look similar to the classical heat equation, but is actually very different. An equivalent and compact notation for this motion is

$$\frac{\partial \mathcal{C}}{\partial t} = \frac{\partial^2 \mathcal{C}}{\partial s^2} = \kappa \vec{\mathcal{N}},$$

where κ is the curvature and $\vec{\mathcal{N}}$ the unit normal. This is a nonlinear equation, called the *intrinsic heat equation* or *mean curvature motion*, and moves each point along the normal direction, an amount proportional to the curvature at that point. Therefore, at points where the curve is locally convex the curve moves inward, and at points locally concave the curve moves outward; asymptotically the shape vanishes as a circle.

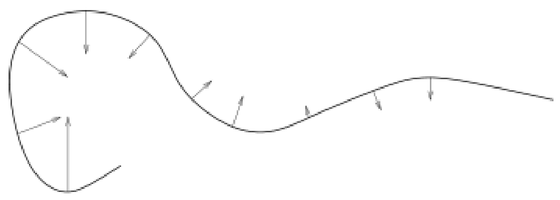

Figure 2.5: Curve with vectors indicating the term $\kappa\vec{\mathcal{N}}$.

We have just pointed out the relationship between the Gaussian filtering and the heat equation. An analogous correspondence can be made between the intrinsic heat equation and the median filter. A precise statement and demonstration can be found in [44].

In certain applications, we may want these motions to be covariant with some geometric transformations. As a part of a more general theory, we remark the invariance under affine transformations preserving area or volume. The motion proposed independently by Sapiro and Tannenbaum ([55], [56]) and Alvarez, Guichard, Lions and Morel [1]

$$\frac{\partial \mathcal{C}}{\partial t} = \kappa^{1/3} \vec{\mathcal{N}},$$

is invariant under the special affine group, and it is called *affine intrinsic heat equation* or *affine curve shortening.*

For the $2D$ case (curves), results about the existence, uniqueness and regularity of the solutions of the mean curvature motion and the affine version exist. However, the general case of motions governed by a generic function of the curvature still have unknown properties.

Let us come back to the $3D$ case, processing volumes and seeing how its level surfaces evolve. In this case, as described in the previous section, two curvatures are needed to describe the the local shape (for instance κ_1 and κ_2, or $\mathcal{H}$ and K), so the range of possibilities is much wider, and the theory is still not completely understood.

The extension of the mean curvature motion is straightforward, and is given by the equation

$$\frac{\partial \mathcal{S}}{\partial t} = \mathcal{H}\vec{\mathcal{N}}, \tag{2.4}$$

analogous to the $2D$ version.

This motion can be obtained from a variational formulation. Namely, if we want to minimize the total area of a surface

$$A = \int\int da \ ,$$

then the corresponding gradient descent flow is the mean curvature motion (2.4).

Nevertheless, this surface MCM version does not behaves as well as for curves. Several examples of topological changes of the surface can be found. For example, a dumbbell (two spheres joined by a cylinder, so it is homeomorphic to a sphere) may evolve into two separates topological spheres [59], see Figure 2.6. However, if the cylinder is thick enough then the surface is homeomorphic to a sphere during the whole evolution process. On the other hand, a torus may evolve into a topological sphere.

The motion by Gaussian curvature

$$\frac{\partial \mathcal{S}}{\partial t} = K\vec{\mathcal{N}}$$

is the next motion that comes to one's mind[6], but it cannot be applied successfully because at non-convex neighborhoods it goes unstable, as shown in Figure 2.7.

A suitable version of the motion by Gaussian curvature is the following:

$$\frac{\partial \mathcal{S}}{\partial t} = (K^+)^{1/4}\vec{\mathcal{N}} \quad \text{where} \quad K^+ = \max(K, 0) \ . \tag{2.5}$$

This motion is the analogous to the affine curve shortening, in the following sense: it is the simplest which admits the affine group (all 3×3 matrices with unit determinant, combined with the translations) as its symmetry group. Caselles and Sbert [11] proved that for this motion, the evolution of the dumbbell does not create singularities (unlike the MCM), so the evolving surface is homeomorphic to a sphere during the evolution.

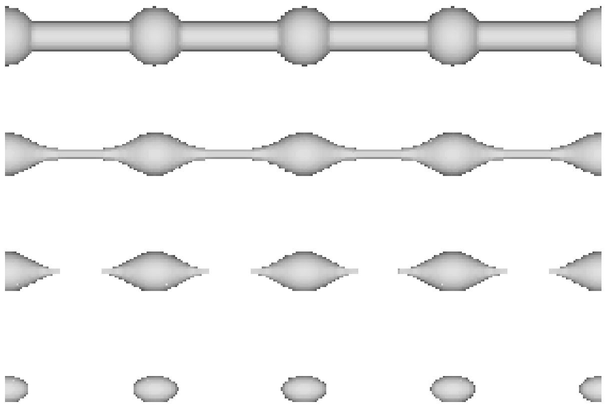
Figure 2.6: Deformation of a dumbbell under motion by mean curvature. Note the topology change. Figure from [16].

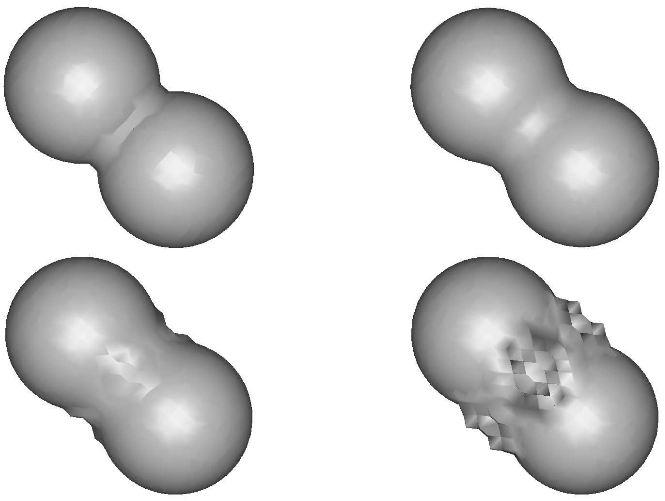
Figure 2.7: Deformation of a dumbbell under motion by gaussian curvature. Note how the evolution goes unstable at the non-convex region. Figure from [16].

However, they also showed that a "bent dumbbell" (still homeomorphic to a sphere) evolves into two separate topological spheres, see Figure 2.8.

The last curvature motion here presented is the one by minimal curvature

$$\frac{\partial \mathcal{S}}{\partial t} = \kappa_{min} \vec{\mathcal{N}} \ .$$

[6]This motion was first proposed in [27] to model the wearing process of a convex stone on the beach.

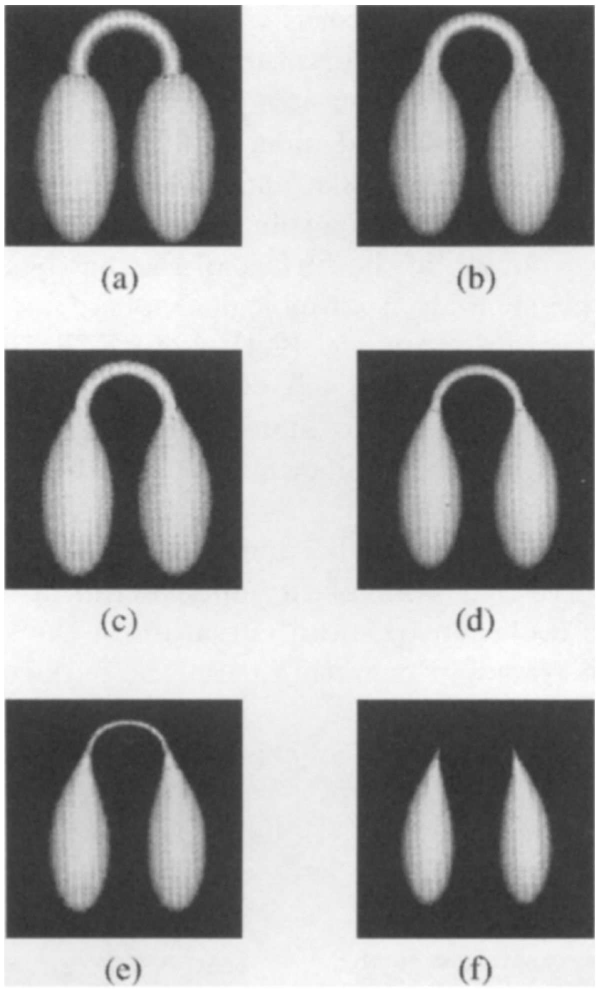

Figure 2.8: Deformation of a bent dumbbell under the curvature motion (2.5). Note the changes in the topology. Figure from [11].

Caselles and Sbert [11] showed with experimental results that this evolution is the only among those just described that preserves the topology of the bent dumbbell (see Figure 2.9). Nevertheless, it is not true that this evolution preserves the topology for every initial surface, a more sophisticated example of a surface that changes its topology during the evolution by minimal curvature is also described in [11].

We remark that all these motions smooth the surface equally at convex and concave shapes, what is clearly desirable in some applications, but it may not be optimal for our problem of detecting a special type of shape. We will come back to this in Section 3.3.

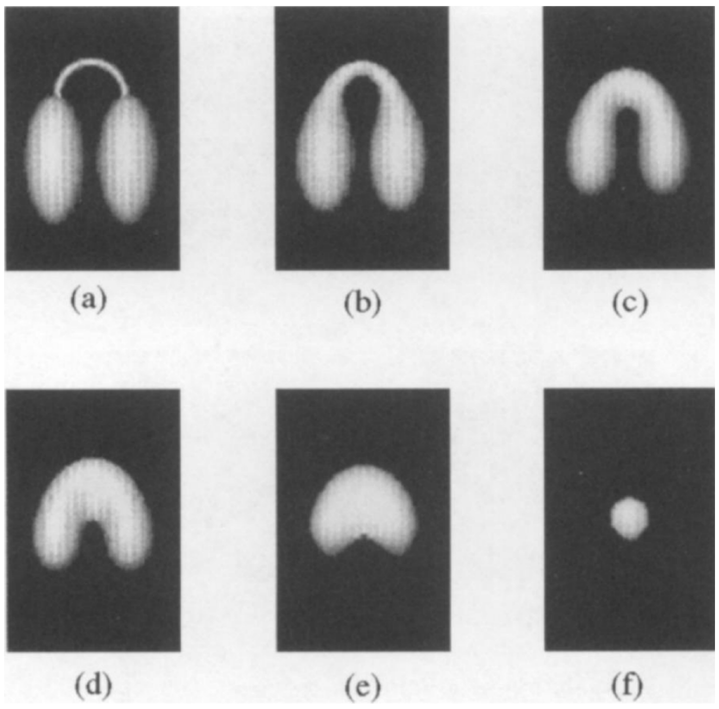

Figure 2.9: Deformation of a bent dumbbell under motion by minimal curvature. This motion preserves the topology in this case. Figure from [11].

2.2.3 Variational formulation - Energy-based snakes

In the previous section we briefly described the fundamental PDE schemes used for smoothing. In the current section we do the same with the segmentation problem, which is roughly, in this case, to locate an object and its boundary (the object will be the colon inside and the boundary the inner colon surface).

All the techniques described in this work seek to minimize a certain functional like

$$E_{\mathcal{S}} = \int_{\mathcal{S}} L(\mathcal{S}) da$$

designed in such a way that the minimum is obtained in the boundary of the object to segment. This optimization problem can be solved by means of gradient descent, resulting in a deforming surface evolving from the initial contour $\mathcal{S}_0$ toward the boundary of the desired object. As we commented above, it is convenient to use the level set method, so we define an equivalent energy defined for u:

$$E_u = \int_{\Omega} L(\mathbf{x})\delta(u)||\nabla u||d\mathbf{x}$$

where Ω is the domain, $\mathbf{x}$ a generic coordinate vector in Ω and δ is the Dirac delta function. Now the corresponding gradient descent of E_u can be computed with standard calculus of variations (see for example [54] [4]), obtaining an Eulerian flow for u instead of $\mathcal{S}$. At this point, the models can be proposed in the surface-energy form, or directly in the level set formulation.

Let us begin with the first *snake* proposed. The classical snake model by Kass et al. [34] was first introduced for curves, and for a parameterized curve $\mathcal{C} : [0, 1] \to \mathbb{R}^2$ (included in an image I) consists of

$$E_{\mathcal{C}} = \alpha \int_0^1 |\mathcal{C}'(p)|^2 dp + \beta \int_0^1 |\mathcal{C}''(p)|^2 dp - \lambda \int_0^1 |\nabla I(\mathcal{C}(p))| dp$$

with α, β, λ positive numbers. While the first two terms favor the smoothness of the resulting curve, the latter attracts the curve toward the object to segment, and the trade-off between these properties is given by the parameters α, β, λ.

This method has several drawbacks, for instance it is not intrinsic and it is very unstable. The improvement by Caselles et al. [9], Geodesic Active Contours, can be interpreted as finding a curve of minimal weighted length. It defines a new Riemannian metric, and the problem is formulated as the minimization of a new length that takes into account the image properties. The energy in this model is

$$E_{\mathcal{C}} = \int_{\mathcal{C}} g(I) ds$$

where ds is the arc-length and g is an edge-detector function, a typical choice is $g(I) = 1/(\varepsilon + |\nabla I|^2)$.

The $3D$ case is analogous [10], and it consists in finding the surface of minimal area where, just like for the $2D$ case, the metric of the area is image-dependent:

$$A_R = \int_{\mathcal{S}} g(\nabla I) da$$

Computing the Euler-Lagrange equation leads to the flow

$$\frac{\partial \mathcal{S}(t)}{\partial t} = \left(g(I)\mathcal{H} - \nabla g(I) \cdot \vec{\mathcal{N}} \right) \vec{\mathcal{N}} \ ,$$

whose Level Set formulation is

$$\frac{\partial u}{\partial t} = g(I)|\nabla u| div \left(\frac{\nabla u}{|\nabla u|} \right) + \nabla g(I) \cdot \nabla u \ .$$

In both equations, the first term favors smoothness, while the second one attracts the surface to the boundary of the object. Note that taking the function g constant, the resulting evolution is the mean curvature motion.

Often, both in the $2D$ and $3D$ case, another force $\nu g(I)$, with ν constant, is added, resulting in the flow

$$\frac{\partial u}{\partial t} = g(I)|\nabla u| div \left(\frac{\nabla u}{|\nabla u|} \right) + \nabla g(I) \cdot \nabla u + \nu g(I)|\nabla u| \ .$$

Unlike the smoothing flows presented in the previous section, where the way to use them is to make evolve the original image a few iterations, in these kind of algorithms the initial condition is an initial contour (actually an image with the initial contour as a level set, usually a distance map to the contour), a sphere for example, and this surface evolves all the way until convergence.

From this point several variants were proposed, adding or modifying terms. We will mention just one direction among them, the Geodesic Active Regions by Paragios and Deriche [50],[51] or the similar model by Chan and Vese: Active Contours Without Edges [13]. In the Geodesic Active Regions, a probability model of the regions is assumed, that is, the probability of a pixel of belonging to a given region, and a term regarding this is added to the energy. The goal is then to minimize the energy

$$\begin{aligned} E &= \alpha \int_0^1 g\left(I(C_1(s))\right) |C_1'(s)| ds + (1-\alpha) \int_{R_1} \log\left(p_1(I(x,y))\right) dxdy \\ &+ (1-\alpha) \int_{R_2} \log\left(p_2(I(x,y))\right) dxdy \end{aligned} \tag{2.6}$$

where $C_1(s)$ is the target arc-length parameterized curve that divides the image domain into R_1 and R_2. The function g is an edge detector, and p_1 and p_2 are the probabilities of belonging to R_1 and R_2 respectively. The first term is the same as in the Geodesic Active Contours model, written slightly different, and the other two terms measure the partition probability, according to the prior probability model for the regions.

The level set formulation of this algorithm is as follows

$$\frac{\partial u}{\partial t} = \alpha(r_2(I) - r_1(I))|\nabla u| + (1-\alpha)\left(b(I)\mathcal{H} - \nabla b(I) \cdot \frac{\nabla u}{|\nabla u|}\right)|\nabla u|$$

where r_2 and r_1 are the region attraction functions (based on the probabilities p_1 and p_2) and b is the boundary attraction function (based on the function g).

This concludes the chapter, where we described all the basic tools used in what follows. These concepts are strongly used in the next chapter, where we tackle the colon segmentation problem.

Chapter 3

Colon Segmentation

3.1 A challenging problem

In this chapter we focus on the colon segmentation problem from CT volumes. The proposed solution is strongly based on the theory of surface evolution briefly summarized in the previous chapter.

In practice, the CT-based colon segmentation problem suffers from several difficulties. Among them we can remark the image resolution, specially taking into account the polyp size that we want to detect. In addition, the presence of liquid, the interface between air and liquid and specially the T-junction formed by air-tissue-liquid, make it hard to obtain reasonable results, and therefore the simple approaches fail. Figure 3.1 shows a detailed view of a slice region, where the different sections can be distinguished and the problem of the air-liquid interface (specially at the T-junction) is clearly evidenced. Figure 3.2 shows a slice and a detailed view of a small polyp which, in addition, is submerged in liquid. Here it can be seen that this polyps is only 4 or 5 voxels wide along its largest dimension.

In addition, there are some desirable properties of the resulting surface, that may not be easy to achieve. For example, as we will compute curvatures for the polyp characterization, the surface must be smooth enough, and it would be nice if the shape of the polyps are enhanced and the slight variations get flattened.

On the other hand, there are a number of desirable properties of the whole process: it should be as fast as possible, automatic, non-parametric and it should work well for all the studies in the database.

The segmentation of the colon surface, which is critical in particular to compute geometric features, is divided into two parts: a pre-processing stage for dealing with the air-liquid composition of the colon volume, and a second stage that consists on smoothing the pre-processed image and obtaining the final colon surface by thresholding the smoothed volume. The overall procedure here presented is very simple and computationally efficient, leading to the state-of-the-art classification results later reported.

In what follows we first describe a preprocessing stage in order to deal with the air-liquid-tissue interface problem, then we study different smoothing schemes of the kind

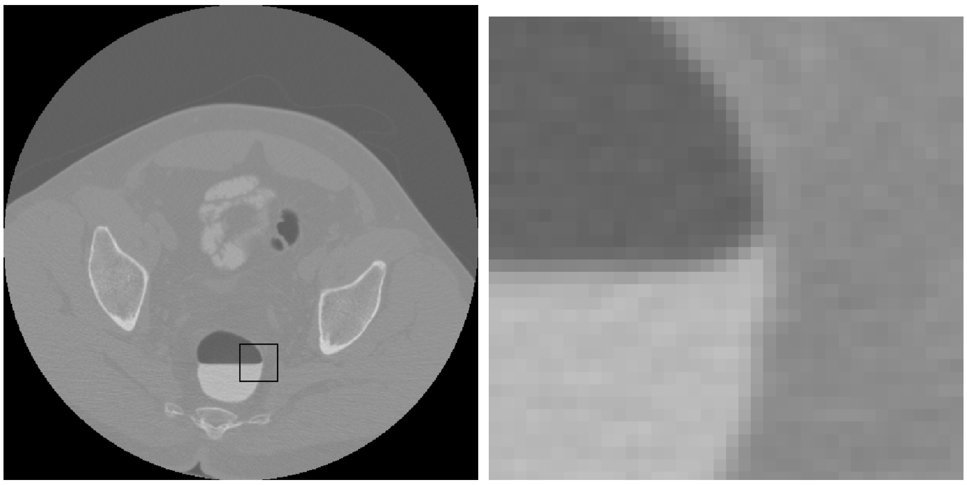

Figure 3.1: CT image and zoom in the air-liquid-tissue interface region.

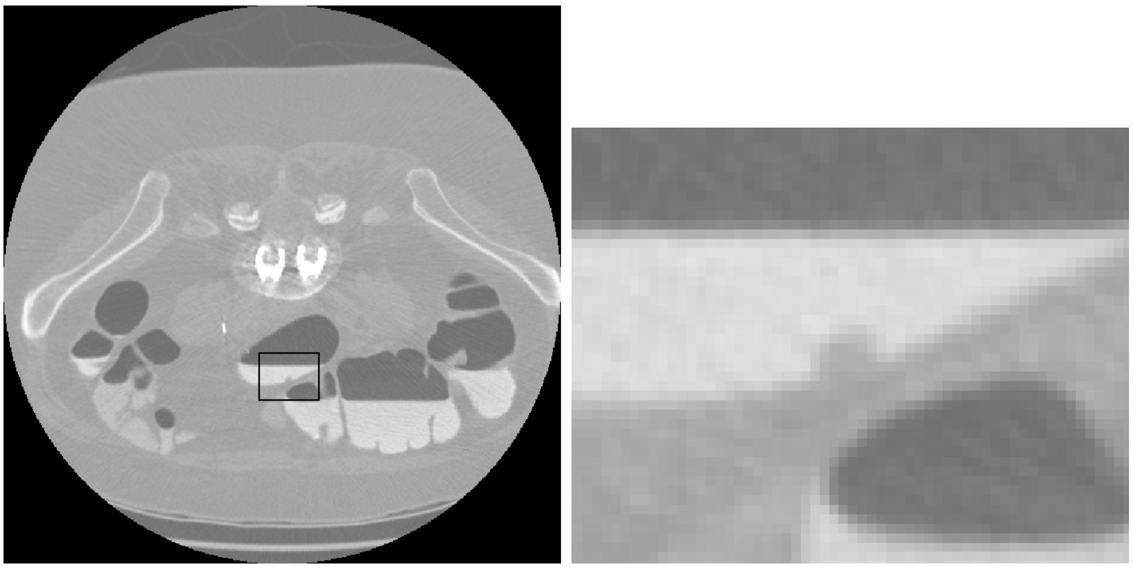

Figure 3.2: CT image and zoom in a polyp submerged in liquid.

presented in Section 2.2.2, and finally we describe two different approaches that proved successful.

3.2 Classifying CT regions

As we just mentioned, one of the strongest difficulties concerning the segmentation of the colon from abdominal CT volumes is the presence of liquid and its interfaces with air and tissue. Figure 3.3 shows a CT slice and its pixel values over the highlighted vertical segment. At first sight there are three clearly distinguishable classes: the lowest gray levels correspond to air, the highest levels correspond to liquid, and the middle gray values correspond to tissue. Nevertheless, there are around 6 interface voxels between air and liquid whose gray values, due to continuity, lie within the normal tissue range. Therefore, a naïve approach based on gray values only, ignoring the physical nature of the tissue and its environment, is not suitable for proper tissue classification and segmentation.

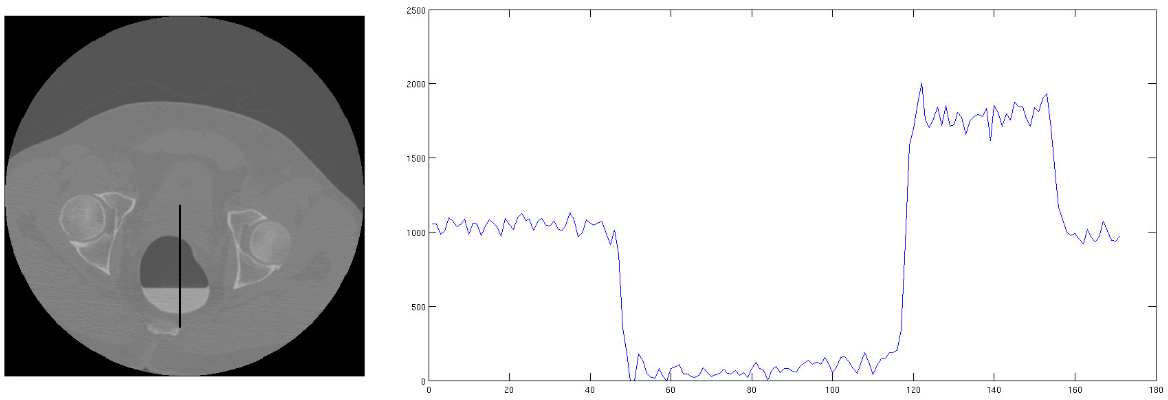

Figure 3.3: CT slice and typical values for air, liquid and normal tissue.

The proposed approach addressing this issue consist on generating a *probability map* **P**, that is, a $3D$ image with the same size of the CT volume (around $512 \times 512 \times 400$ for our data), whose values are the (estimated) probability of the voxel of being inside the colon. We will then process this map to estimate the interior colon wall.

We obtain **P** by computing for each voxel the three probabilities of belonging to each of the three classes of interest (namely, air, liquid and interface, the union of these three classes forms the inside of the colon), and then taking **P** as the maximum of these three probabilities.

Since the gray value distributions of air (class w_1) and liquid (class w_2) are very distinguishable and stable among the different studies, they can be empirically learned by manually segmenting these two classes in a given CT study, and then constructing the probability distribution functions $p(x|w_1)$ and $p(x|w_2)$ by standard kernel density estimation [72]. From this, the distributions $p(w_1|x)$ and $p(w_2|x)$ can be computed, assuming uniform class priors. The air and liquid probabilities for a voxel in the testing CT are then easily computed by simply evaluating these estimated distributions on the gray value of the CT voxel.

The challenging component of computing **P** is the computation of the interface probability (class w_3). Here we take advantage of the physics of the problem, and in particular of the gravity and the position of the patient: the person is laid horizontally so the interface between the liquid and the air is a plane parallel to the floor. The voxels situated on the interface then have a large gradient in the z (vertical) direction, since the values of air and liquid are on opposite ends. However, the transition is about 6 voxels wide for the standard data resolution used in this work, and the computations should be done taking this into account. Additionally, if a given voxel belongs to the interface layer, it is expected that at least half of the neighbor voxels at the same horizontal plane also belong to the interface layer.

The implementation of these criteria is as follows. A cubic neighborhood around each voxel $\mathbf{x}$ is considered, and for each one of the "columns" that result of fixing the x and y coordinates, the probabilities of the upper voxels of being air and the lower voxels of being liquid are accumulated. If the tested voxel belongs indeed to the interface layer, then all these air and liquid probabilities will be high. The interface probability $P(w_3)$ is then an increasing function of this accumulated measures. The algorithm in Figure 3.4 provides a pseudo-code that represents this procedure.

```
for each voxel (x,y,z) do
    sum=0;
    for $i$=−1 to 1 do
        for $j$=−1 to 1 do
            for $k$=1 to 2 do
                sum += $p(w_1|$(x+i,y+j,z+k));
                sum += $p(w_2|$(x+i,y+j,z-k));
            end
        end
    end
    $p(w_3|$(x,y,z)) = sum/18;
end
```

Figure 3.4: Interface probability computation.

Now, let us comment a couple of things about the other interfaces, namely, air-tissue and liquid-tissue, which are the most important ones if we want to delimit the colon wall. First, the voxels of these interfaces are not confused with the air-liquid interface computed by the algorithm presented above, because the value of the gradient in the z direction is, at most, approximately half of gradient value in air-liquid voxels. Indeed, the jump from air to liquid is around 1900 units, while the jump from air or liquid to tissue is around 1000. On the other hand, the intensity value of these interface voxels has mainly two contributions, the one of the tissue and the one of the air (or liquid). This value is then somewhere in between 0 (air) and 1000 (normal tissue), for example. Therefore, the air-liquid interface probability $p(w_3)$ is low because the gradient value is low, and the liquid probability $p(w_2)$ is low because the distribution of liquid voxel values is centered in 1800 approximately; so the maximum probability (between $p(w_1), p(w_2)$ and $p(w_3)$)is the corresponding to w_1 (air) and the value of the probability map will be $\mathbf{P}(\mathbf{x}) = p(w_1|\mathbf{x})$, as expected. The value of this probability depends on the contributions of each region (air and tissue), varying from 1 (only air) to almost 0 (all tissue). Therefore, the probability map $\mathbf{P}$ vanishes smoothly at the actual border of the colon.

The voxels on the T-junction (air-liquid-tissue interface) have high interface probability $p(w_3)$ because they have air above and liquid below. Nevertheless, there are two factors that decrease this probability. The first one is due to the fact that, within the considered neighborhood in algorithm 3.4, there are some normal tissue voxels, so the accumulated variable *sum* is not as high as in the case of a "pure" air-liquid interface. The second factor is due to the surface tension, that causes that the angle between the wall and the liquid is not exactly a right angle, and therefore the hypothesis of horizontal interface is slightly violated. With the contribution of these factors, the probability $p(w_3)$ is lower than one, and vanishes smoothly at the T-junction.

After the computation of the probability map some spurious (isolated) voxels may have high probability of being liquid (bones for example, or simply noise), so we clean the probability map by keeping the connected components[1] containing some chosen voxels used as seeds. The seeds are automatically detected choosing the voxels with greatest values of $P(w_3)$, since these high probabilities occur only at the true interface between air and liquid. This way, the segmentation step is able to handle the lumen discontinuities problem and obtain the multiple pieces of the colon that might be disconnected.

Figure 3.5 shows a slice of the original volume data and the same slice of the computed probability map $\mathbf{P}$.

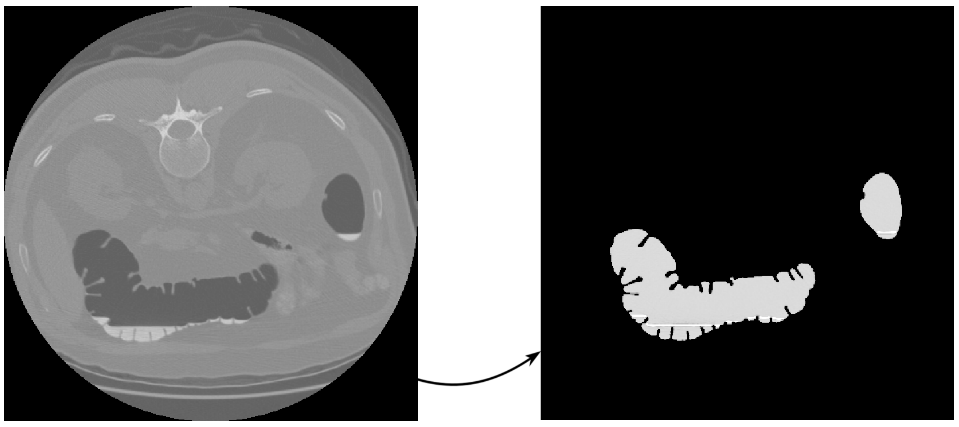

Figure 3.5: CT slice and its corresponding slice on the probability map.

This way of preprocessing the volume leads to a better segmentation compared to the given surfaces we started with, specially at the T-junction zone. Figure 3.6 shows the detail of that zone and the improvement due to the proposed probability map. It can be seen how the artifact produced by the interface, the "gutter" shape, is considerably reduced. This improvement is critical, since if small oscillations occur along the "gutter" (which is expectable given due to the resolution of the CT image) artifacts with a polyp-like shape are produced, and this, of course, degrades the performance of the whole CAD system.

[1]Actually, since the probability map is not binary, a (conservative) threshold of 0.6 is considered to separate the connected components.

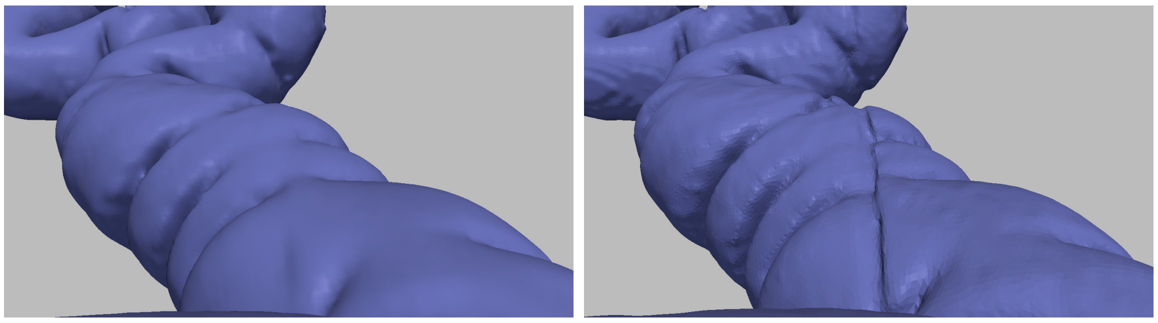

Figure 3.6: Comparison of our segmentation (left) with the previously provided segmentation. Note how the "gutter" effect is diminished in our segmentation.

3.3 Smoothing

In order to eliminate noise and to obtain a smoother surface after the segmentation, the image should be processed. Part of the theory concerning this subject was presented in Section 2.2.2. Here we study these techniques and propose some improvements for our particular application.

One of the fundamental aspects that we look for in this smoothing process is the preservation of the shape of the polyps, while obtaining a smooth enough surface to reliably compute geometric features such as curvature. Of course, the ultimate goal is to derive a method to process the surface that simplifies and improves the polyp/non-polyp classification system.

At this point we have the previously computed probability map $\mathbf{P}$, which will be processed by a Partial Differential Equation of the form

$$\frac{\partial u(\mathbf{x},t)}{\partial t} = \beta|\nabla u| \quad , \quad u(\mathbf{x},0) = \mathbf{P}(\mathbf{x}) \ , \tag{3.1}$$

where $\mathbf{P}$ is the initial condition. After making a few iterations of this PDE evolution, an iso-level surface of the resulting $3D$ image $u(\mathbf{x},T)$ will be extracted, and that will be the resulting colon surface. Since we are interested in a certain iso-level surface of the volume, it will be useful to recall the relation that states how the iso-levels evolve when the $3D$ image evolves according to (3.1). Namely, the basic equation of the Level Set Method [48] states that if $u(\mathbf{x},t)$ evolves according to (3.1), then its iso-levels (level sets) satisfy

$$\frac{\partial \mathcal{S}}{\partial t} = \beta\vec{\mathcal{N}}, \tag{3.2}$$

where $\mathcal{S}$ is any iso-level surface and $\vec{\mathcal{N}}$ its unit normal.

Therefore, in what follows we study surface motions, in particular when the deformation term β is a function of the principal curvatures κ_{max} and κ_{min}. In the actual implementation, this term (and therefore the principal curvatures) have to be computed from the function $u(\mathbf{x}, t)$, a standard procedure in the implementation of curvature-based surface motions [47]. We first review the classical motions by curvature and then propose some variations used as part of our developed pipeline.

The classical mean curvature motion

$$\frac{\partial \mathcal{S}}{\partial t} = \mathcal{H}\vec{\mathcal{N}}$$

has problems regarding topological changes, as we discussed in the previous chapter. Nevertheless, making only a few iterations this problem does not affect our surface. With very few iterations a very smooth surface is obtained, but the polyps are flattened fast too, as it can be observed in Figure 3.7. However, if used carefully, it is a very good alternative to more standard Gaussian filtering.

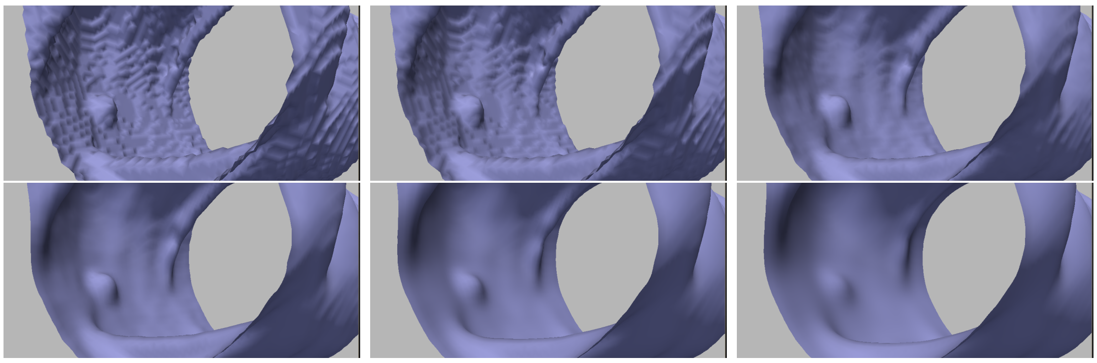

Figure 3.7: Mean Curvature Motion: original surface and the result after 2, 8, 15, 30 and 50 iterations. Note how both the surface (as desired) and potential polyps (undesired) are smoothed and flattened.

We have already seen that the motion by Gaussian curvature has several problems with surfaces containing non-convex parts, which is our case. Indeed, only a few iterations are needed for the result to explode (decompose into disconnected parts), as shows Figure 3.8.

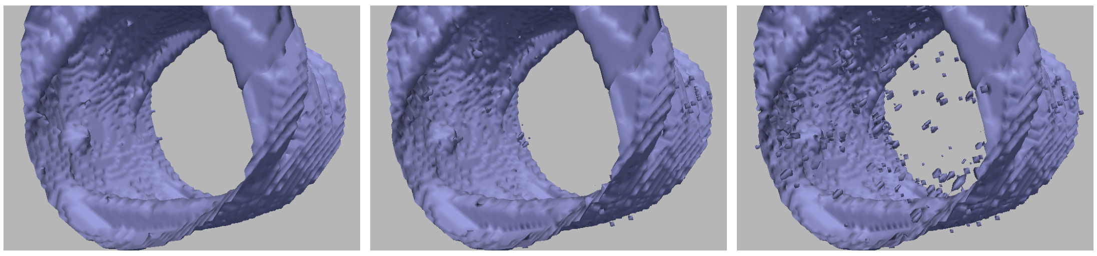

Figure 3.8: Gaussian Curvature Motion: surface after 1, 3 and 4 iterations.

On the other hand, the affine motion

$$\frac{\partial \mathcal{S}}{\partial t} = (K^+)^{1/4}\vec{\mathcal{N}} \quad \text{where} \quad K^+ = \max(K, 0)\ ,$$

has a better behavior in general, in terms of topology preservation. However, when used with only a few iterations, the results are comparable with the mean curvature motion.

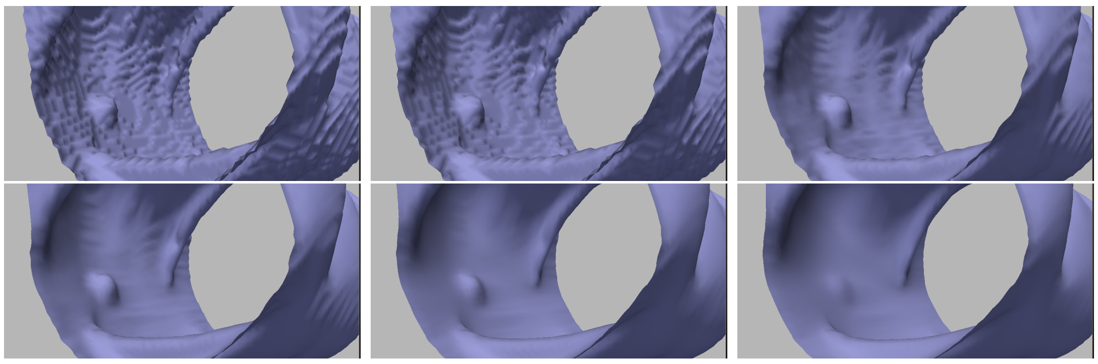

Figure 3.9: Affine Gaussian Curvature Motion: original surface and the result after 2, 8, 15, 30 and 50 iterations.

The last classical motion presented in Section 2.2.2 is the motion by minimal curvature κ_{min}. Polyps have a curve of inflection points all around it, separating the upper and lower sectiong of the polyp. Along this curve, the minimal curvature is $\kappa_{min} = 0$, and therefore this section of the polyp does not move (or moves very slowly), so intuitively under this motion the polyps should persist longer. In our particular application, this evolution yields very good results in terms of both surface smoothing and polyp enhancement (see Figure 3.10).

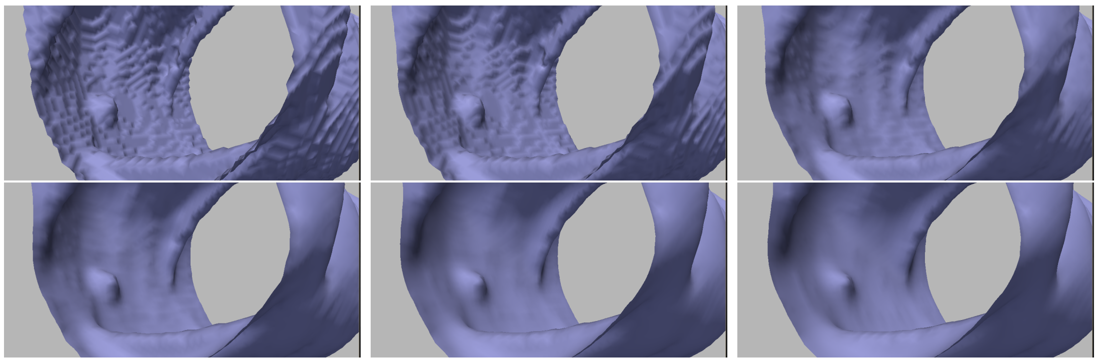

Figure 3.10: Motion by minimal curvature: original surface and the result after 2, 8, 15, 30 and 50 iterations.

In what follows we try to take advantage of the known properties of our shapes of interest to propose motions that behave better in our application. Inspired in the exponent 1/4 of the affine motions in dimension 3, and the best (in terms of polyp-shape preservation) of the motions presented above, we first tested the evolution that obeys the equation

$$\frac{\partial \mathcal{S}}{\partial t} = \kappa_{min}^{1/4} \vec{\mathcal{N}} \ .$$

It turns out that the motion governed by this equation yields better results than the ones we have just presented. Figure 3.11 shows the result after a few iterations, and Figure 3.12 evidences the difference with a comparative image: the result of the motion by κ_{min} is in gray while the result of the motion by $\kappa_{min}^{1/4}$ is shown in orange. It can be seen that the polyp surface in the latter is above the polyp surface of the former, and the surrounding zone is the other way around, showing that the evolution by $\kappa_{min}^{1/4}$ better enhances the polyp when compared with the motion by minimal curvature.

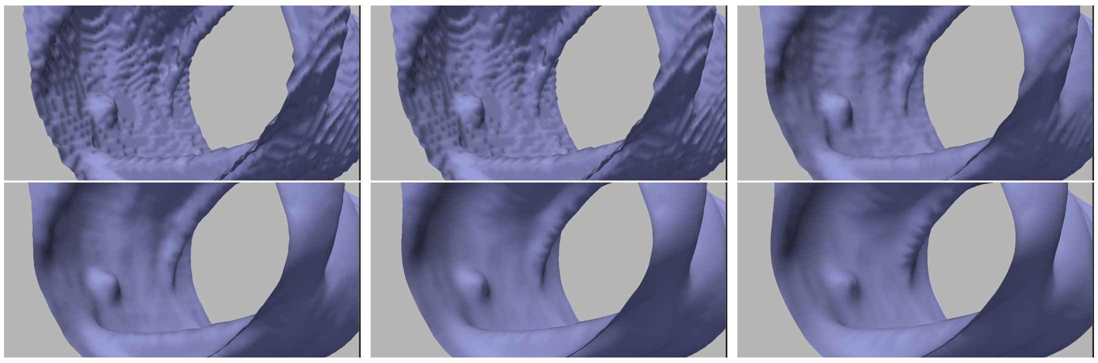

Figure 3.11: Evolution by $\kappa_{min}^{1/4}$: original surface and the result after 2, 8, 15, 30 and 50 iterations.

In order to further improve the smoothing process, we will characterize the polyps shape and distinguish even more the evolution for those zones.

The next step is to include this information concerning the shape of the surface in order to make potential polyps evolve differently than the rest of the colon surface. One option is to start from the motions studied above and make slight modifications to the functions to take into account the Koenderink curvature measures. Another option is to propose a completely new function, being careful so that the resulting evolution is stable. We tested some motions depending only on the shape index, but with no better results than the motion by $\kappa_{min}^{1/4}$. However, may be worth a deeper study of these motions.

On the other hand, we can try to modify the best motion so far ($\kappa_{min}^{1/4}$), in such a way that the resulting motion further enhances the potential polyps. To achieve this, we need to characterize the potentially polyp points and modify the deformation function according to that. The shape index characterization is very precise, but the computation of the shape index of the level surface from the embedding function u is delicate. The technique described in Section 2.2.1 (the eigenvalues of the matrix in equation (2.3)) gives accurate results, so using this curvature computation the shape index characterization is precise.

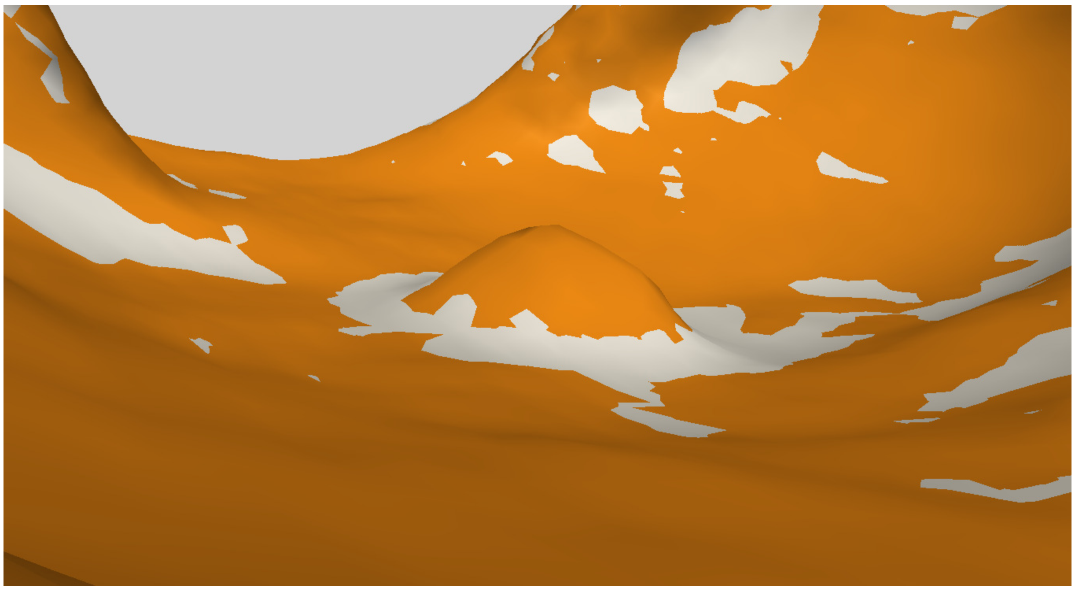

Figure 3.12: Comparison between evolutions. Motion by k_{min} in gray vs. motion by $k_{min}^{1/4}$ in orange, 15 iterations in both cases.

We now define a function that acts as a multiplying factor of the term $\kappa_{min}^{1/4}$, making the surface evolve slower at the points of interest. One option is to choose this function to depend on the shape index only, assigning low values to shape index near -1, and values close to unity to other points. A smooth function $g(SI)$ verifying these constraints is shown in Figure 3.13.

The final motion then becomes

$$\frac{\partial \mathcal{S}}{\partial t} = g(SI)\, \kappa_{min}^{1/4} \vec{\mathcal{N}} \ .$$

This proposed evolution keeps all the advantages of the motion by $\kappa_{min}^{1/4}$ and in addition, the polyps are flattened slower, so at the end the obtained surface is smooth and the polyps are still outstanding. The stopping time T was manually chosen at 15 iterations. This choice has to be made once for all, as the acquisition framework is kept unchanged.

At this point, after choosing the appropriate diffusion, we have a smoothed probability map $u(\mathbf{x}, T)$ indicating the volume inside of the colon. At least two paths arise now in order to obtain the colon surface: segment the smoothed probability map $u(\mathbf{x}, T)$ with a snake technique or directly threshold $u(\mathbf{x}, T)$ with some criteria. These approaches are discussed in the following sections.

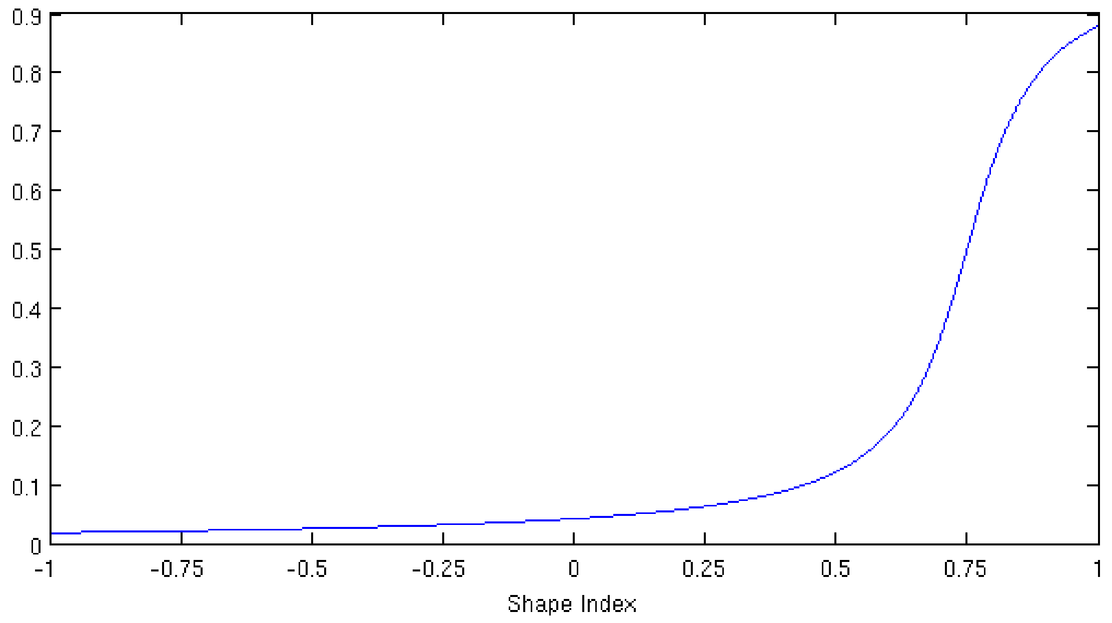

Figure 3.13: Function $g(SI) = \frac{1}{\pi}\arctan\left((SI - 0.75)\cdot 10\right) + \frac{1}{2}$, multiplying factor for PDE curvature evolution.

3.4 Colon surface extraction

3.4.1 Thresholding P - Marching Cubes

The main advantage of this scheme is the simplicity and the speed. Basically, using the Marching Cubes algorithm a certain iso-level surface is extracted from the probability map.

The Marching Cubes [40] is a well-known algorithm that, given a volume and a value, returns the level-surface of the given value with sub-voxel accuracy. In this application, a value $\alpha \in [0, 1]$ is chosen, as the image is a probability map. The algorithm works as follows. For each cube formed by eight contiguous voxels, it is determined how the sought surface intersects the cube, this polygon (formed by triangles) is stored and the algorithm proceed with another cube. The final surface is obtained by gluing these triangles.

Given a cube, the vertices are grouped into two classes, those whose value is greater and lower than the given threshold. Then, according to the resulting pattern of the vertices grouping (which, up to rotations and inversions is one of the 15 possible), the decision of where does the surface intersects the cube is taken from the table in 3.14. The sub-voxel accuracy is obtained by interpolation. This is, given an edge where the surface should cross, the exact location of the intersection is obtained by linear interpolation along the edge.

The choice of the value α can be made by maximizing some criteria, in order to obtain the best surface in a given sense, based on contrast for instance like the work by

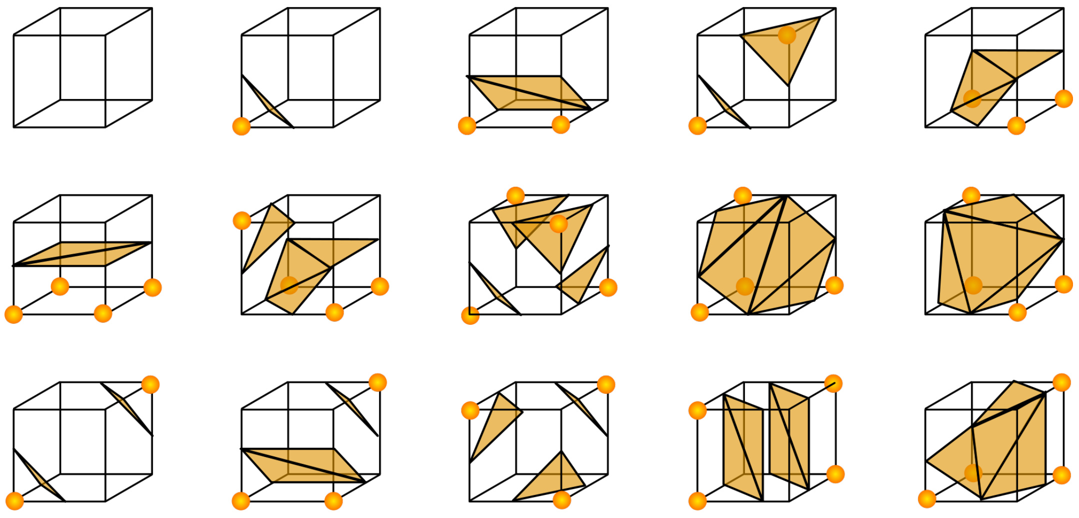

Figure 3.14: Marching cubes: the 15 possible configurations.

Meinhardt et al [42]. This optimization-oriented method was tested, but in our particular application all the consistent surfaces are very close to each other (see Figure 3.15), and all of them are reasonable segmentations of the colon. Therefore, the computational effort is not justified and we simply use a fixed value, extracting the iso-level surface for $\alpha = 0.7$. Note that this choice can be safely made once for all, since all the studies are performed under very similar conditions. The result of this stage is then a triangulated surface $\mathcal{S}$ representing the colon wall.

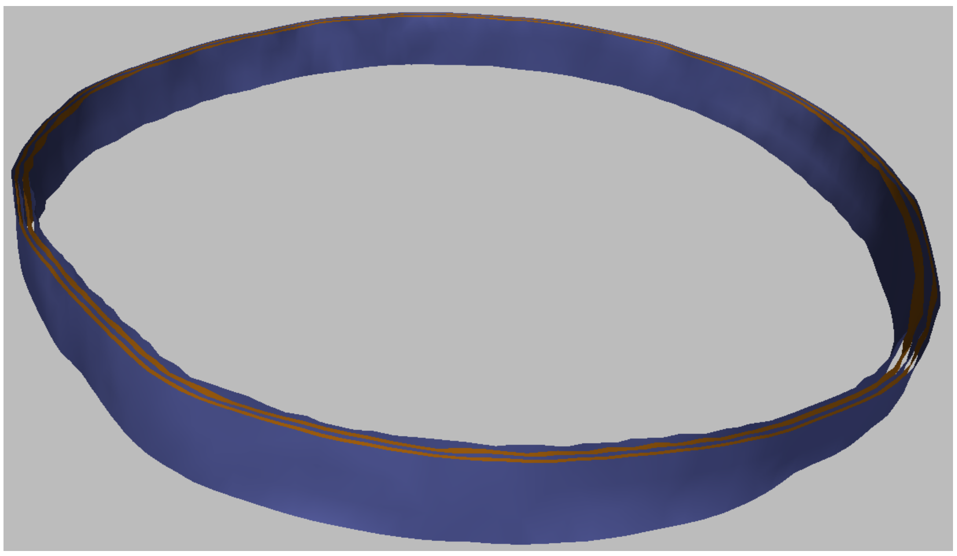

Figure 3.15: Different iso-level surfaces: values 0.5,0.6,0.7,0.8 and 0.9.

3.4.2 Surface evolution techniques - Geodesic Active Regions

Among the active contours methods that were reviewed in Section 2.2.3, the Geodesic Active Regions is the surface evolution technique that is best suited for our problem. Indeed, in addition to the surface regularity term of the snakes model and the image gradient information added by the GAC method, GAR allows to incorporate information about the colon interior and exterior regions. The main ideas of the Geodesic Active Regions algorithm were presented in Section 2.2.3. Basically, there are three images to choose: the region attraction functions r_i and the boundary attraction function b. The latter is chosen as the classical edge detector function $\frac{1}{\varepsilon+|\nabla\tilde{u}|^p}$, where $\tilde{u}$ is an smoother version of u and $p \geq 1$. The other term is actually the difference between the region functions r_2 and r_1. In Section 3.2 we described the computation of a probability map, which complies with all the requirements for being the region function r_2, indicating the probability that a voxel belongs to the inside of the region to segment: the inside of the colon. Although we do not have an analogous function r_1 for the background, the complement of r_2 can be used, that is, $r_1(x) = 1 - r_2(x)$. With this setting and a reasonable set of parameters, the algorithm is able to correctly segment the colon surface, starting from any initial contour.

These kind of methods are very useful when there is no precise idea of where the object to segment is, so the boundary attraction terms forces the evolving surface to converge to the object while keeping a certain degree of smoothness due to the regularization terms. But in our case, the probability map is quite descriptive of the colon volume, and the boundaries of these images (the original CT volume and the probability map) are not sharp enough to make the attraction extremely accurate. So the boundary attraction term is not crucial, and the smoothing terms are equivalent to those studied in the previous section. Indeed, if we choose the particular function $g \equiv 1$, the resulting flow is the mean curvature motion, which is one of the first diffusion techniques discussed before. In practice, for these particular images, the value of the parameters needed to obtain a smooth image are such that the polyps result very flattened. To illustrate this fact, Figure 3.16 shows in orange the smoothest segmentation that can be obtained with GAR, while not flattening too much the polyp in Figure 3.12. It is clear that surface is too rough to compute curvatures and its derived geometrical features. Also, it is worth to say that the final surface is obtained by applying the Marching Cubes, in order to obtain the zero level set of the evolved function, so the difference between the two schemes regarding the smoothness of the final surface relies on the differences between the smoothed probability map and the level set function of the active contour techniques. Taking into account these facts and, again, the search for speed and simplicity, we decided to use the approach described in Section 3.4.1, namely to directly threshold the regularized probability map and to obtain the colon inner surface with a marching cubes procedure.

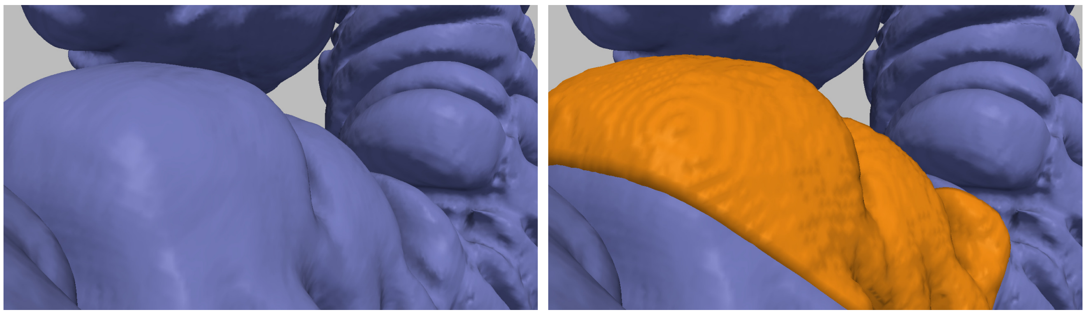

Figure 3.16: Segmentation using the proposed method (left) and using GAR (right, in orange).

3.5 Summary

In this section we give a brief description of the whole segmentation step. The input is the $3D$ CT volume (around $512 \times 512 \times 400$), and the first stage is to compute the probability map $\mathbf{P}$, which indicates the estimated probability that each voxel belongs to the inside of the colon. This is achieved by computing probabilities of being air, liquid or air-liquid interface voxels. The maximum of these probabilities is taken, and a connectivity filter is used to get rid of the spurious sections with high probability. Then this probability map $\mathbf{P}$ is processed with a PDE. Namely, $\mathbf{P}$ is the initial condition of the following equation:

$$\frac{\partial u(\mathbf{x},t)}{\partial t} = g(SI)\,\kappa_{min}^{1/4}|\nabla u| \quad , \quad u(\mathbf{x},0) = \mathbf{P}(\mathbf{x}) \ .$$

After 15 iterations of this motion, a smoothed probability map $u(\mathbf{x},T)$ is obtained. Finally, an iso-level surface of this $3D$ volume is extracted by using the marching cubes algorithm, with a iso-level value of 0.7. We insist that these two parameters are the only parameters involved in the whole segmentation stage, and can be safely chosen once for all.

Chapter 4

Classification

So far we have addressed the segmentation part with the corresponding preprocessing stage. In this chapter we describe the second fundamental block of the pipeline: the classification stage, including the candidate patches detection and the feature analysis. We also describe the class imbalance problem and how we dealt with it, and we analyze the classifiers that were trained for polyp detection.

The input data for this part is the original CT volume and the segmented mesh obtained with the technique described in the previous chapter. The goal is to output the polyp zones with the fewest number of false positive as possible. Of course, the major objective is not to miss true polyps.

Briefly, the whole process carried out in this chapter consist of three stages. The first one is the detection of the regions (or patches) of the segmented surface $\mathcal{S}$ that are candidates of being polyps, this includes the exact delineation of the region, which is achieved by a multi-scale search (testing several sizes and keeping the most appropriate one). In the second step, for each candidate patch several features are computed. Geometric features are computed from the surface $\mathcal{S}$ and texture features are computed from the original $3D$ CT volume data. The majority of all these features take into account the information of the surrounding area of the candidate patch. In the final step, a classifier takes as input these features and decides if the candidate patch is indeed flagged as polyp or non-polyp.

A very brief and basic description of the classification framework is given in what follows. A dataset composed of two classes is given, in our application the classes are *Polyp* and *Non-Polyp*. The dataset contains *instances*, which are vectors x belonging to one of the classes and whose entries are *features*. In short, the dataset $\mathcal{D} \subset \mathbb{R}^n$ is partitioned in classes $\mathcal{D} = A \cup B$ with $A \cap B = \phi$. A *classifier*, after being trained, decides to which class x belongs from the values of the features. Sometimes it is useful to label the classes with $+1$ and -1 for mathematical purposes. A typical and very simple classifier is shown in Figure 4.1.

If a particular instance x_0 is used both for the training and testing of the classifier the evaluation of the performance is not fair. In an ideal framework, with a large enough dataset, one should divide the dataset into training and testing datasets. However, as it is the case here, the number of instances in one or both classes is not large enough.

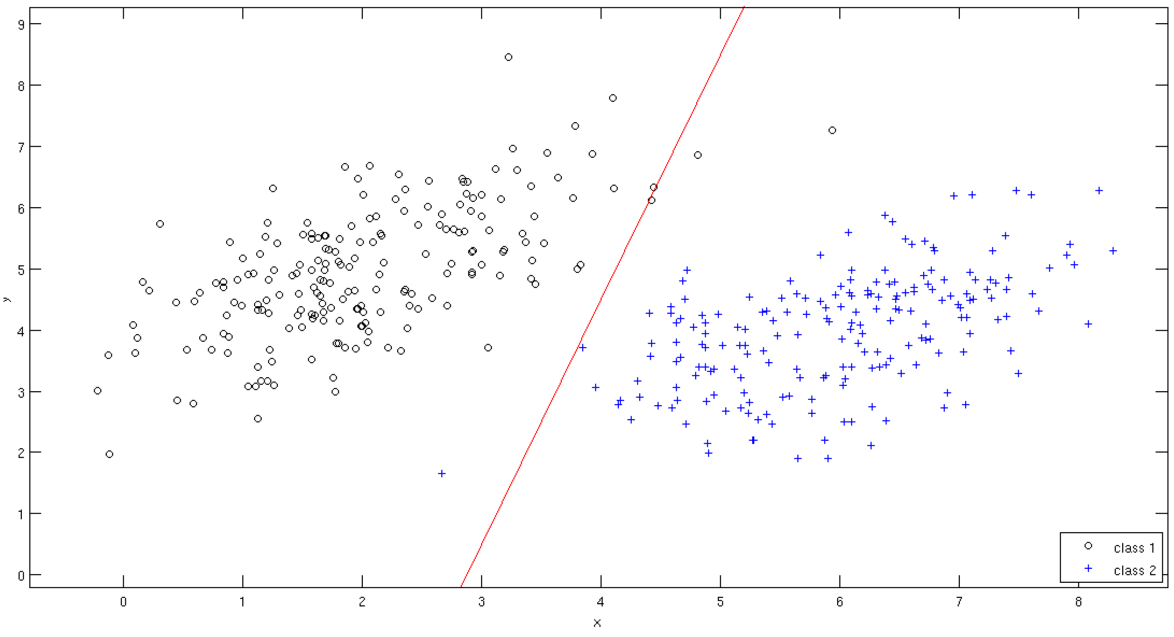

Figure 4.1: Typical linear classifier.

Under these circumstances, the process is carried out by a scheme called *cross validation*: the dataset is partitioned into n folds, then $n-1$ folds are used for training and the other one for the testing. This procedure is repeated n times, rotating the testing fold, and then a performance measure is taken from the n experiments. The described technique is a particular case called *n-fold cross validation*, we use a slightly different variant, explained in Section 5.1.3.

Several measures are defined in order to characterize the performance of the classifiers. The basic measures are the True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN), almost all the other measures are derived from these ones. The TP are the true polyps correctly classified, while the FP are zones that the classifier decides that they are polyps but they are not. Of course, the goal is to have no FN (that is, all the polyps correctly detected) while keeping a reasonable amount of FP. In this application, it makes no sense to use measures based on the TN, because the amount of instances in the Non-Polyp class is dependent on the algorithm (the instances to classify is the output of the candidate detection stage). The sensitivity, however, is defined as $\frac{TP}{TP+FN}$, so it depends on the number of true polyps only, and it measures the percentage of polyps correctly detected. Therefore, we will evaluate and compare the different schemes with the sensitivity and the number of False Positives (actually, the number of FP per patient study).

Given a classifier, different operating points can be chosen, for example, sacrificing the number of false positives for the sake of obtaining a high true positive rate. In general, this variety of TP-FP relation can be achieved by moving a parameter, a decision threshold for instance. The most used graphic representation of the different operating points is the ROC (Receiver Operating Characteristic) curve of a classifier. This graph plots the true positive rate versus the false positive rate. There are two points that trivially belongs to every ROC curve: $(0, 0)$, which means that no true positives were classified and all the negatives were correctly classified, this is achieved by classifying every instance as negative. On the other hand, classifying every instance as positive gives a true positive rate of 1, but also a false positive rate of 1. The ROC curve joining these two points gives a performance measure of the classifier, and the operating point in this curve is chosen according to the particular nature of the application.

Since the measures based on FN or TN should not be used to describe the performance of a polyp CAD algorithm, as it was said, the false positives per study is the common used quantity. In this sense, the analogous to the ROC curve is the FROC curve, which plots the sensitivity or true positive rate versus the false positives per study (usually the average over all the available studies).

4.1 Candidate detection and feature computation

4.1.1 Candidate detection and delineation

The first step is to define a set of candidate patches over the mesh $\mathcal{S}$. Since the obtained mesh has about a million points, it is completely unfeasible to consider them all, so we take advantage of the particular geometry of the polyps to reduce the amount of candidates to test. Let us consider the shape index as a function $SI : \mathcal{S} \to [-1, 1]$, and recall that the polyps have shape index values close to -1. Therefore, it is expected that a region (patch) of the surface that corresponds to a polyp contains at least one local minimum of the shape index function. The detection of the candidate patches starts from this observation, and follows a multi-scale search. For each local minimum $x_0 \in \mathcal{S}$ of the function SI, several level sets of SI $(\mathcal{P}_1 \ldots \mathcal{P}_n)$ around x_0 are tested, and the level set $\mathcal{P}_i$ that maximizes the distances between the SI histograms of the region and its surrounding, described below, is the final considered candidate patch, which we simply denote by $\mathcal{P}$. A total of $n = 7$ level sets are tested, corresponding to the shape index values from -0.8 to -0.5 with a -0.05 step. These different patches $\mathcal{P}_1 \ldots \mathcal{P}_n$ are illustrated in Figure 4.2. On the one hand, values of shape index around -1 are expected in the central part of the polyp (if the acquisition and the segmentation are reasonable). On the other hand, a shape index around -0.5 means a valley, which is what we expect at the boundary of the polyp, in its joint with the normal tissue. See Figure 4.3. Different polyps have different shapes and joints with the normal tissue, the maximization that gives the final path intends to capture the polyp structure as good as possible.

The description in Section 4.1.3 of the mentioned SI histograms is given for the final chosen patch $\mathcal{P}$, but the ring and histogram computations are made for all the level sets $\mathcal{P}_i$ in order to select the most appropriate of them (the most appropriate scale). This multi-scale delineation is one of the contributions of this work.

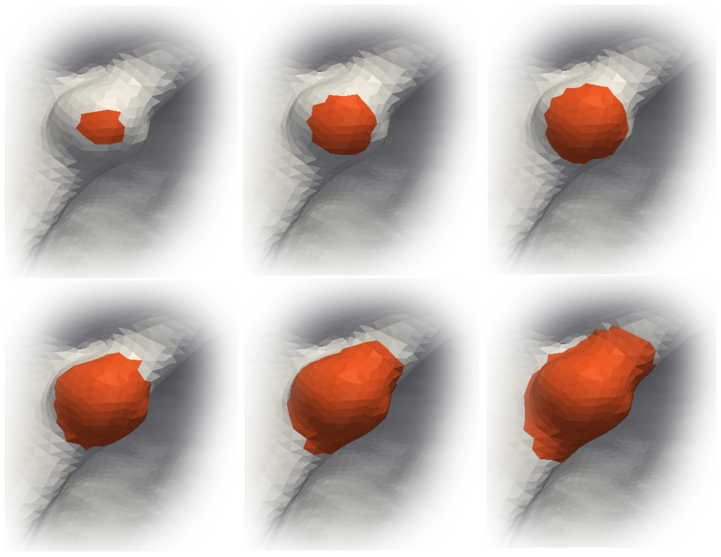

Figure 4.2: Different patches $\mathcal{P}_1 \ldots \mathcal{P}_n$ are tested, in order to determine the size that best fits the polyp candidate region.

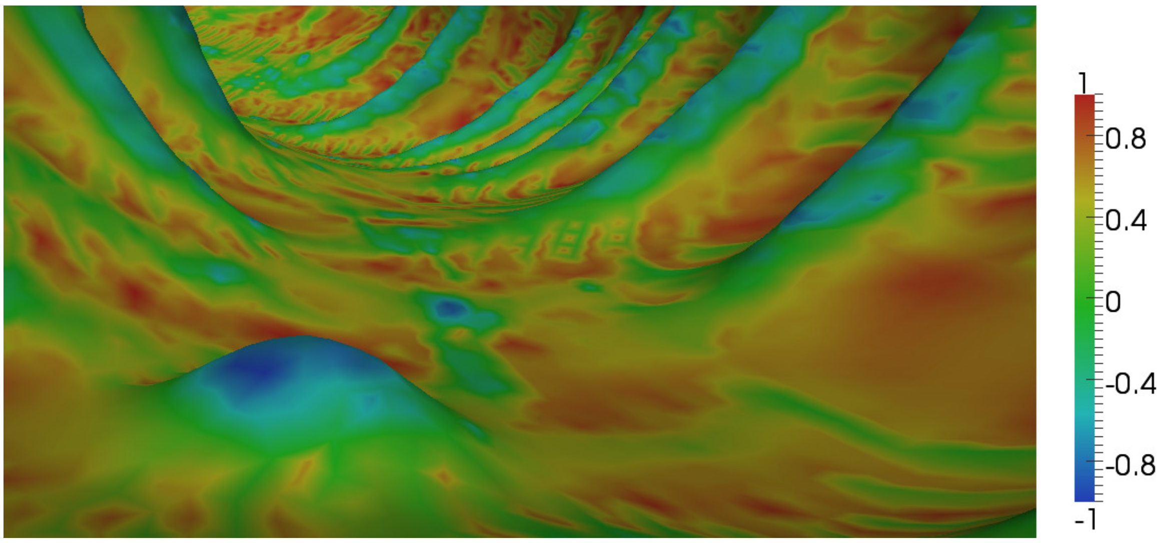

Figure 4.3: Shape Index values over the surface. Note how the values along the polyp are concentrated around -1.

4.1.2 Intuition in context information

All the polyp detection methods reported in the literature try to detect or classify the polyps from properties defined only within the candidate region, without considering the data around the region. However, it is important to analyze the context in which the candidate patch is located, not only because different sections of the colon present different characteristics, but also because polyps can be situated over different structures such as folds or plain colonic wall. A good feature who includes the shape of the neighborhood for example, can help in the discrimination between irregular folds and polyps over folds. Besides that, the fact of looking for significant differences in the gray level, as we detail in Section 4.1.4, tries to imitate the human eye behavior, which highlights zones that contrast with their vicinity.

In this regard, most of the features described in this Section take into account the local information of the area around the candidate patch. Polyps (actually all the candidate patches) are then characterized not only by their intrinsic geometry and structure, but also by their relationship with the surrounding area.

This, in a certain sense, makes the measures independent from the particular phenomena going around, in a context where the natural variations of the properties of the colon tissue impact on the measures and make absolute thresholds nonsense.

4.1.3 Geometrical features

In this part we describe how a polyp should look like from a geometrical point of view, and how to take advantage of this description in order to distinguish between *Polyps* and *Non-polyps*. We will see that, although there are different types of polyps, classified by their shapes and joint with the colon tissue, there are some geometrical properties in common that allow us to characterize all of them in terms of the curvatures presented in Section 2.1.2.

We recall that polyps can be classified into Flat, Sessile and Pedunculated. The three of them are supported in a circular region, and are more or less pronounced. With the chosen normal orientation, this implies values of shape index indicating a shape like a spherical cap (roughly, values between -1 and -0.6). The size of the polyps can cover a wide range of values, for example in the database where we obtained the test cases there are polyps from $2mm$ to $50mm$. Nevertheless, as the shape index is scale-invariant, while the shape of the polyp is spherical-like, the shape index will lie within the mentioned interval; the curvedness on the other side, is the one indicating the "scale" and it will serve to distinguish between artifacts and the shapes with the size we are looking for (that is, polyps larger than $3mm$ for example).

The different types of polyps have different joints with the colon wall: while sessile and flat lesions have a smooth transition, pedunculated polyps are attached by a stalk. This produces different shape index values along the border of the lesions, but the multi-scale strategy is able to capture the whole polyp correctly.

In the spirit of including the context information as explained in Section 4.1.2, we propose the measures described below, taken from regions defined as follows. We note that in this section all the sets are defined over $\mathcal{S}$, because the geometric measures are made directly from the mesh surface.

Given a candidate patch $\mathcal{P}$, a ring $\mathcal{R}$ around $\mathcal{P}$ is computed, in order to consider geometrical measurements with respect to the area surrounding the patch. The ring is calculated by dilating the patch $\mathcal{P}$ a certain geodesic distance, such that the areas of

$\mathcal{P}$ and $\mathcal{R}$ are equal. The geodesic distance computation on the triangulated surface is performed using the algorithm in [36].

This geodesic computation algorithm works, in short, as follows. Given a point $\mathbf{p}$ in the mesh $\mathcal{S}$, the geodesic distance to the other points in the mesh is computed. For the neighbors of $\mathbf{p}$ (the points sharing an edge with $\mathbf{p}$) the distance is simply the length of the edge. The set of points where the distance is already computed (the *Alive* set) is incremented at each step with the points one edge away of the *Alive* set (the *Close* set). The computation for each one of these points is made considering the triangle that it forms with the points of the *Alive* set, and then solving a trigonometric equation. We refer to [36, 5] for more details.

Figure 4.4 shows a candidate patch (actually a true polyp), and its corresponding ring.

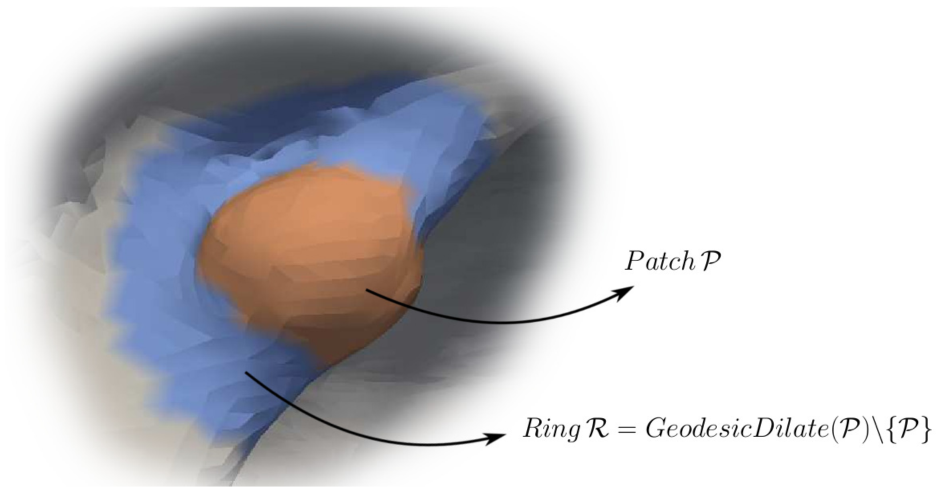
$Patch\,\mathcal{P}$
$Ring\,\mathcal{R} = GeodesicDilate(\mathcal{P})\backslash\{\mathcal{P}\}$

Figure 4.4: Ring (in blue) surrounding a candidate polyp (in orange).

Histograms of the shape index values are then computed for the patch $\mathcal{P}$ and the ring $\mathcal{R}$, and two different distances between them are computed: the L_1 distance and the symmetric Kullback-Leibler divergence (see Figure 4.5). If the patch corresponds to a polyp-like shape then the values of the histogram $\mathcal{P}$ will be concentrated around the -1 extrema, on the other hand, the histogram $\mathcal{R}$ will be inclined to the other extreme in case of a polyp on a normal colon wall (concave), or with tendency to values near -0.5 if the polyp is on a fold. These two features give a measure of the geometric local variation of the candidate patch $\mathcal{P}$. We assume that there are no other polyps in $\mathcal{R}$ or that they do not significantly affect the statistics on the ring.

Although these two are the most discriminative features, we also consider the following additional ones since they still help to discriminate some typical false positives:

- The mean value of the shape index over the patch $\mathcal{P}$, which describes the shape of the selected patch.

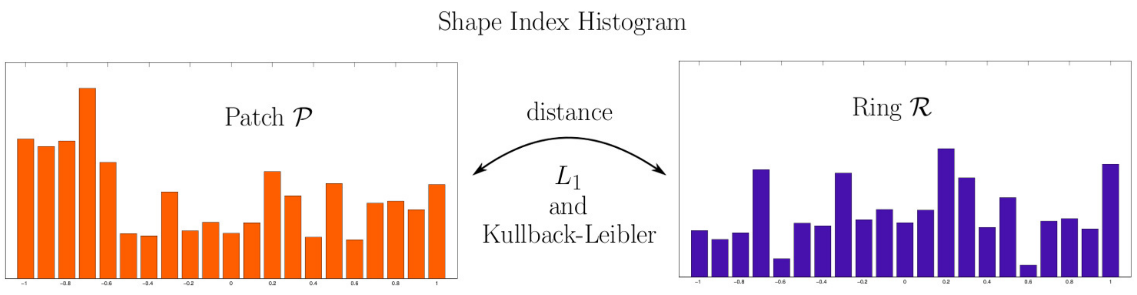

Figure 4.5: Typical shape index histograms and distances between them.

- The area of the patch, since we want to detect polyps in a certain range of sizes.
- The growth rate of the areas at the multi-size stage, meaning the ratio between the area of the chosen patch $\mathcal{P} = \mathcal{P}_i$ and the area of the immediately smaller tested patch $\mathcal{P}_{i-1}$; this feature measures how fast the shape of the patch is changing, in a context where it is difficult to quantize the variation of the shape.
- Finally the *shape factor*

  $$SF = \frac{4\pi \cdot Area}{Perimeter^2} \quad ,$$

  which measures the shape of the patch border, how efficiently the perimeter is used in order to gain area,[1] and it favors circle-like patches (like the polyp patch in Figure 4.4), avoiding elongated patches (like the patches in folds, which are a very common source of false positives).

We then end-up with a total of 6 geometric features for detecting candidate polyps, namely: L_1 and Kullback-Leibler distance between shape index histograms of patches and corresponding rings, the mean shape index over $\mathcal{P}$, the area of the candidate patch, the growth rate of the areas and the shape factor.

4.1.4 Texture features

Having ended the geometrical description of the polyps with the corresponding features, we move to the other type of properties that will feed the classification stage, these are the texture and gray level features. In what follows in this section, all the values are taken, of course, from the CT volume, because the gray levels (absorvance values) are in the original CT image.

It is quite obvious that the geometric factors are very important and decisive. The influence of the texture or gray level, on the contrary, may be less intuitive. Due to the differences in biological activity of polyp cells, the gray-level in the polyps may show differences with the normal tissue, as can be seen in Figure 4.6, and it can be

[1]The maximum value for the shape factor is 1 and its achieved only by the circle.

very helpful for detecting polyps. This is in particular useful for flat or small polyps, where the geometric information is limited. Besides the gray-level itself, the texture of the lesion may look suspicious, so we also describe in this section the texture features studied during the course of this work.

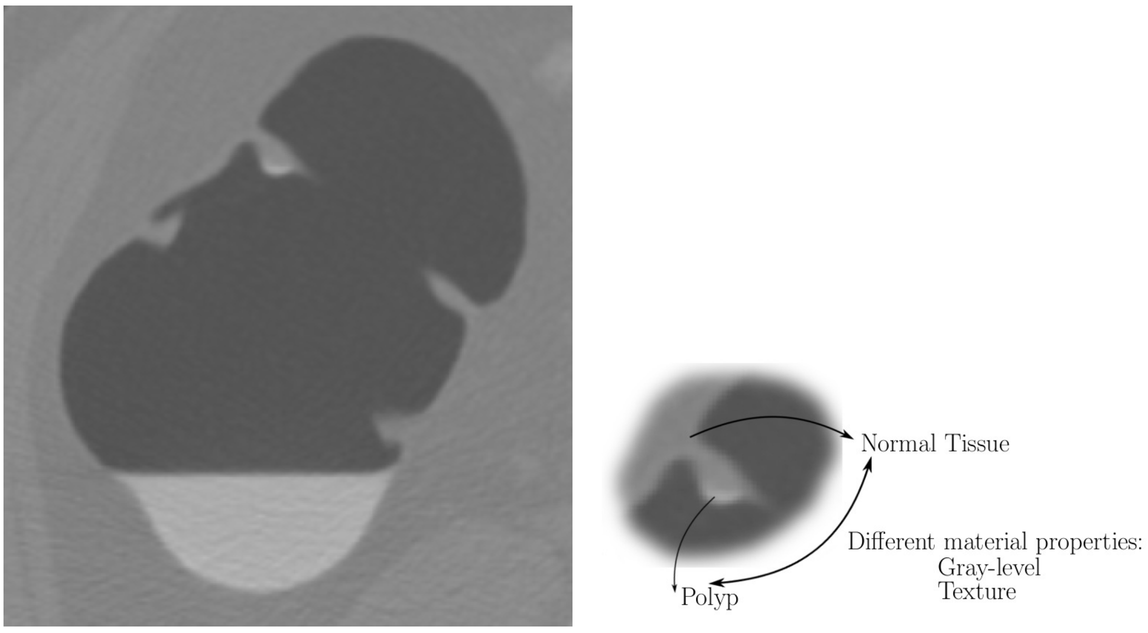

Figure 4.6: Gray level difference between polyp and normal tissue.

Some work has been done on the inclusion of texture features (computed inside the candidate polyps only), in order to reduce false positives [73], but according to the reported results, there is still a lot of room for improvement in texture features. In [73], the authors propose to compute, for each voxel inside the candidate volume, the three eigenvalues of its corresponding Hessian matrix, and they claim that the distribution of these eigenvalues for polyps voxels differs from the distribution for non-polyp voxels. We tested these proposed features, but we did not find them correlated with the class.

Here we propose both the use of new texture features and the inclusion of the information on the candidate's surrounding area, as described in Section 4.1.2.

In this regard, again, we first define a couple of sets. First, for each polyp candidate $\mathcal{P} \subset \mathcal{S}$, a volume $\mathbf{P}$ is calculated, containing the patch $\mathcal{P}$ and a portion of the inner tissue next to the patch. This volume $\mathbf{P}$ is intended to contain the polyp tissue, so just voxels with gray-level within a certain range (avoiding air and liquid) are considered. A second volume $\mathbf{R}$ surrounding $\mathbf{P}$ is calculated, in order to compare it with the polyp candidate tissue. We expect $\mathbf{R}$ to contain normal tissue. $\mathbf{R}$ is computed by dilating the volume $\mathbf{P}$ and avoiding the air and liquid voxels, this is illustrated in Figure 4.7.

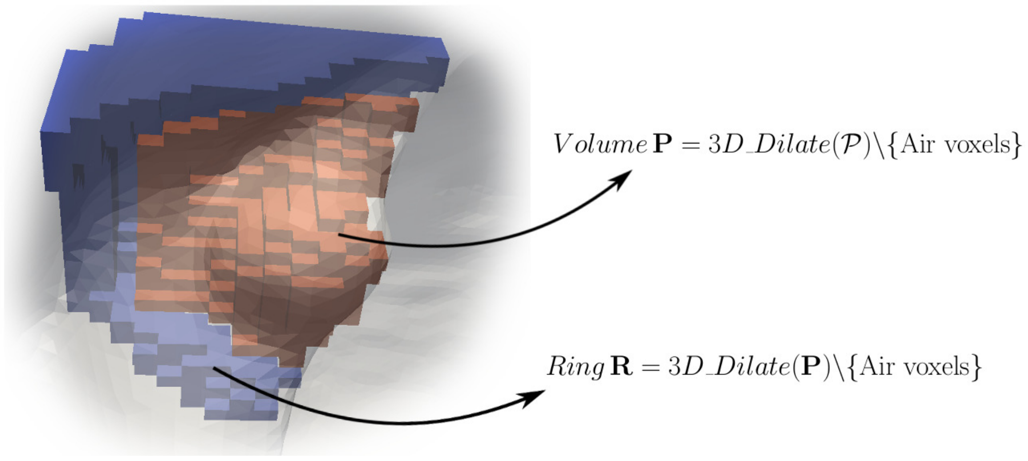
$Volume\,\mathbf{P} = 3D\_Dilate(\mathcal{P})\backslash\{\text{Air voxels}\}$
$Ring\,\mathbf{R} = 3D\_Dilate(\mathbf{P})\backslash\{\text{Air voxels}\}$

Figure 4.7: Volumes **P** and **R** for texture features computation.

Haralick features

One of the chosen features is the mean gray level, and the others are a subset of the classical Haralick texture features [31], adapted to the $3D$ case. In the original paper Haralick describes co-occurrence matrices for each one of the four direction of a $2D$ image, namely, horizontal, vertical and the two diagonals. The row and column number of these matrices index all the gray-level values, and the i, j entry of the matrix of a given direction counts the number of times gray level values i and j have been neighbors in that direction. The proposed features are then derived from these matrices, which describes the spatial dependence of the gray levels of the image.

In the discrete $3D$ case there are more directions to consider[2]. We consider just seven directions, namely $(1,0,0), (0,1,0), (0,0,1), (1,0,1), (1,1,0), (0,1,1)$ and $(1,1,1)$, and the co-occurrence matrices for these angles are calculated with the voxels of **P**, on the one hand, and with the voxels of **R** on the other hand.

For each of the seven directions, six features are computed. The notation will be:

- $p(i, j)$: the normalized i, j entry of the co-occurrence matrix
- N: number of gray-levels

and the six features are the given by the following expressions:

- Entropy:

$$-\sum_{i=1}^{N}\sum_{j=1}^{N} p(i,j)\log p(i,j)$$

[2]Actually in the continuous case there are the same number of directions: uncountable infinity.

- Energy or Angular Second Moment:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} p(i,j)^2$$

- Contrast:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} (i-j)^2 p(i,j)$$

- sumMean:

$$\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} (i+j) p(i,j)$$

- Maximum probability:

$$\max_{i=1\ldots N, j=1\ldots N} p(i,j)$$

- Homogeneity:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{p(i,j)}{1+|i-j|}$$

All the seven features (gray-level plus six Haralick features) are averaged over the seven directions, for each volume **P** and **R**, and the differences between the features over the two volumes is considered (Figure 4.8). For the purpose of comparing the features, those ones computed on **P** only are called *absolute features*, and the differences between **P** and **R** are called *differential features.*

This approach for computing the texture features, measuring differences with the surrounding area, leads to better discrimination than the features computed just for volume **P**, as discussed further in Section 5.2.

4.1.5 Feature Selection

Among all the geometric and texture features considered here, some of them have better discrimination power than others. Some of them, when included in the decision process, may even deteriorate the overall performance of the classifier.This is the main reason why a feature selection step is usually performed.

However, in general an exhaustive search is infeasible, because the number of subsets of a set of n features is 2^n, so heuristic methods are used to find a reasonable solution. Typically, there are two kinds of algorithms: feature ranking and subset selection. The former rank the features according to a given metric, so the features can be sorted. The latter vary the subset (adding or deleting features) according to some heuristic, until reaching an optimal classification performance in a given sense.

Both techniques, feature ranking and subset selection techniques, were applied here, using the implementation from the software package *Weka.*

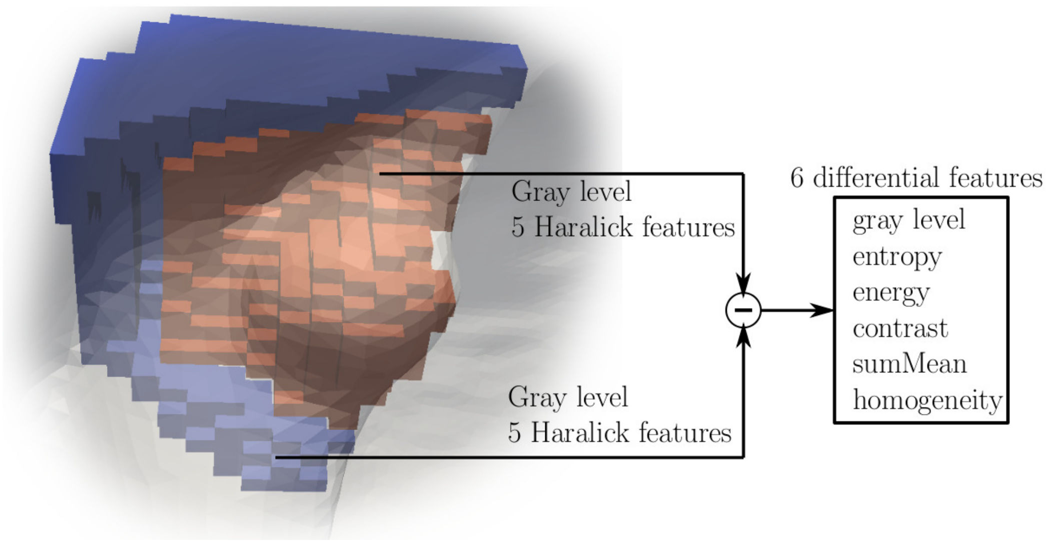

Figure 4.8: Differential texture features.

For the feature ranking, three different single-attribute evaluators (methods to evaluate the features individually) were used, namely, *InfoGainEval*, *GainRatioAttributeEval* and *ReliefFAttributeEval*. The two first ones uses the concept of *entropy*, which is a function originally introduced by Shannon [61] that measures the uncertainty or information of a random variable. The entropy H of a discrete random variable X that takes values in $\{x_1 \dots x_n\}$ is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log(x_i) \ .$$

The most known example is the entropy of a binary random variable with distribution $(p, 1-p)$ (Bernoulli). Figure 4.9 shows the entropy of this random variable, and illustrates the fact that the uncertainty is maximum when the two results are equiprobable ($p = 1/2$), and minimum at the ends ($p = 0$ and $p = 1$), when only one result is possible and therefore there is no uncertainty about the result. It can be also interpreted as "how much information the result really contains".

Furthermore, given two random variables X, Y, the *conditional entropy* $H(X|Y)$ is defined as

$$H(X|Y) = \sum_{i,j} p(x_i, y_j) \log\left(\frac{p(y_j)}{p(x_i, y_j)}\right) \ ,$$

and it measures the amount of uncertainty of X given that you know the value of Y. Finally, the *mutual information* $I(X;Y) = H(X) - H(X|Y)$ measures how much information does Y give about X, or equivalently, how much does Y decreases the uncertainty about X.

Back to the attribute evaluators, the *InfoGainEval* measures the information gain

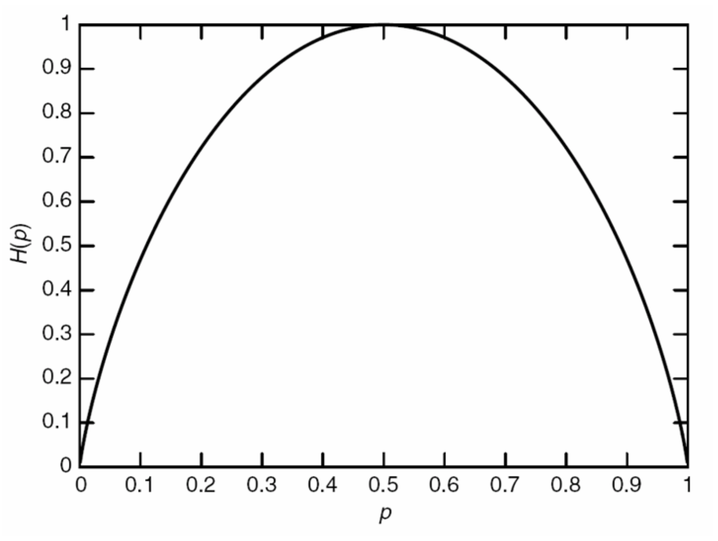

Figure 4.9: Entropy H of the distribution $(p, 1-p)$.

of an attribute with respect to the class:

$$InfoGain(Class, Attribute) = H(Class) - H(Class|Attribute)\ .$$

The *GainRatioAttributeEval* is similar:

$$GainR(Class, Attribute) = \frac{H(Class) - H(Class|Attribute)}{H(Attribute)}\ .$$

The third attribute evaluation technique is the so-called *ReliefFAttributeEval*, which for each instance $\mathbf{x}$ selects the nearest instance $\mathbf{x}^+$ of the same class and the nearest instance $\mathbf{x}^-$ of the opposite class. With this triplet, for each attribute a weight is computed as $w_i = ||x_i - x_i^+||^2 - ||x_i - x_i^-||^2$. This process is repeated a number of times and these weights are updated accumulating w_i. These weights are then used as the relevance measure of each feature.

All these methods are quite unstable: the results (ranking or best subset) vary significantly depending on the heuristic technique used to evaluate the attributes.

However, some general comments can be given. For example, all the rankings obtained with these three different evaluators agree that the geometric features are better ranked than the texture features. Another important comment is that the differential features are always better ranked than the absolute features, this result is further confirmed with the evaluation in Section 5.2.

On the other hand, to select the best subset of features a *wrapper* approach was used, which means that the performance of the classifier itself is used to measure the discrimination power of the subset. Theoretically, this process should lead to the best possible classification results. Nevertheless, because of time constraint reasons, we did not used our evaluation scheme in this wrapper procedure, but the built-in solution in *Weka* which uses a performance metric different than ours.

The number of selected features in the resulting subsets varies from 8 to 11, depending on the greedy search method. However, when we evaluated the performance of these subsets in terms of sensitivity vs false positives, we found that using all 12 features yielded to better results. Indeed, in our setting we seek to minimize the false positive rate under the constraint of 100% sensitivity. However the wrapper technique implemented in *Weka* uses more standard performance measures (Area Under the ROC Curve, Precision, Recall, F-score, etc).

4.2 The Class Imbalance problem

After the candidate detection with the multi-size approach, the number of true polyps was much lower than the number of no-polyps patches, a relation on the order of 500 : 1, which is a huge problem for the learning stage of the classifier. This situation is known as *class imbalance*. In this section we briefly explain the problems encountered when dealing with a class imbalance situation. Then we analyze three solutions and comment on the correct way to use them.

When the dataset is highly imbalanced, the classifier tends to assign the instances to the majority class. A numerical example with our numbers is quite illustrative. If the classifier simply tries to minimize the number of missclasified instances, then the simplest classifier, assigning the majority class always, reaches a 99.8% of well classified instances (all the non-polyps patches). Nevertheless, this classifier misses all the true polyps!

It is obvious that the cost for us to miss a true polyp is much higher than the cost of labeling a non-polyp as a polyp. This can be represented by a cost matrix like

$$\begin{pmatrix} 0 & N \\ 1 & 0 \end{pmatrix}$$

where $N \gg 1$ is the cost of classifying a polyp as non-polyp, 1 is the cost of classifying a non-polyp as polyp, and the cost of classifying correctly is zero.

To fix ideas, let us consider the case with two classes and one feature with Gaussian distribution $\mathcal{N}(0,1)$ and $\mathcal{N}(3,1)$ for classes w_1 and w_2 respectively, as illustrated in Figure 4.10. If the matrix cost is trivial, according to the Bayesian decision theory it follows that the decision rule is to select the class w_1 if the feature value is lower than 1.5 and the class w_2 otherwise.

Now, if we consider a nontrivial cost matrix, say

$$\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} \tag{4.1}$$

then it is easy to show that the optimal decision threshold moves and we decide w_1 if

$$\frac{p(x|w_1)}{p(x|w_2)} > \frac{c_{12} - c_{22}}{c_{21} - c_{11}}$$

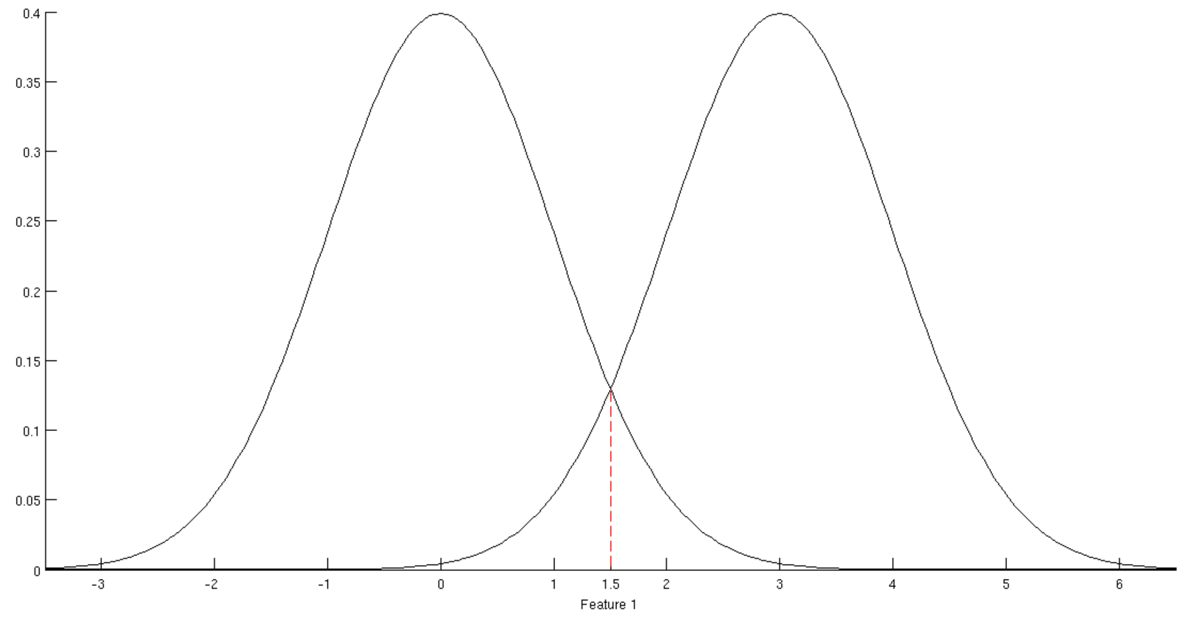

Figure 4.10: Optimal decision rule for two classes with one feature with Gaussian distribution $\mathcal{N}(0,1)$ and $\mathcal{N}(3,1)$.

and w_2 otherwise.

Although it is very simple to modify the threshold for this case, it is not so simple with more sophisticated classifiers. The nontrivial cost matrix in the case of our application and the highly imbalanced data are the problems we addressed with the techniques described in what follows.

4.2.1 Meta Cost

The approach of Domingos [20] consists of combining several instances of the classifier instead of stratification (modify the proportion of classes in the training data according to the costs). This technique is called MetaCost and, in short, it forms different sets from the training data by taking samples with replacement, then the base classifier is trained with each of these sets. When classifying an instance, the probability to belong to each class is estimated by the fraction of votes received from the group of classifiers, then the instance is classified according to the matrix cost minimizing the conditional risk[3]. Because of its working basis, this method does not work with "stable" classifiers (those that produce similar models with slightly different training sets) like SVM or Naive Bayes. Since we chose SVM as basis classifier, our results with MetaCost were not the best.

[3]The conditional risk $R(i|p)$ is the expected cost of predicting that p belongs to class i: $R(i|p) = \sum_j P(j|x)C(i,j)$ where $P(j|x)$ is the probability of x belonging to class j and $C(i,j)$ is the entry of the matrix cost.

4.2.2 Cost Sensitive Learner

This approach, unlike the MetaCost, tries to balance the classes before the learning stage. The implementation we used from *Weka* simply takes as input the re-balance parameters and replicates instances of the minority class. More specifically, the algorithm is set up by a cost matrix like the one in (4.1). Usually, the entries c_{11} and c_{22} are set to zero, because there is no cost associated to correctly classify an instance. The effect of the cost matrix is actually defined by the ratio between c_{12} and c_{21}, so the cost matrix can be represented, without loss of generality, as

$$\begin{pmatrix} 0 & N \\ 1 & 0 \end{pmatrix} . \tag{4.2}$$

With this cost matrix, the minority class (here the polyp class) is over-sampled by replicating N times each instance.Notice that this technique can be applied also in a setting where the classes are balanced, but the costs of missclasification are very different. Therefore, the model trained with the new dataset is inherently less likely to make expensive mistakes.

One of the advantages of this approach is that no assumptions are made about the behavior of the classifiers (unlike the previous method) nor the distribution of the data (unlike the next method).

4.2.3 SMOTE

The *Synthetic Minority Over-sampling TEchnique* (SMOTE) is a method to generate artificial instances of the minority class, in order to get a balanced data to learn from. The new artificial instances are created as a linear combination of the existing instances of the minority class. More specifically, for each new synthetic instance to create, N original instances are randomly selected, and the new artificial one is computed as a convex combination of these N instances. The parameter N is configurable. Therefore, there is an underlying assumption that the optimal partition in the feature space gives convex sets, which may not be the case in several applications. Here is the basic usage: testing with a cross-validation strategy, take the 9 training folds, apply SMOTE to them and use the resulting data for train a classifier, then test it with the remaining fold. It is worth pointing out that the synthetically increased data should be used only to train the classifier and never to test it, because if so the results will be altered, as it can be seen from the following experiment that we made.

We generated artificial data: two classes with five features with random values following a Gaussian distribution $\mathcal{N}(0, 1)$ for both classes (so actually there is only one class). We then applied SMOTE to the whole data and classified it with Naive Bayes using cross validation. Values of Area Under the ROC Curve of almost 0.8 were reached when an area of 0.5 was expected, as the features were independent and identically distributed.

4.3 Classifiers

In this last section of the chapter we describe the different tested classification techniques. All these algorithms are *supervised*, which means that a training data set is used in the learning stage, producing a model or classifier. This model is then used to classify the new instances. The way of splitting the data into training and testing datasets is described in the next chapter, along with more specific details about the election of some parameters.

4.3.1 Support vector machines (SVM)

In this section we describe the foundations of the SVM classifiers family [17]. The basic idea is to represent the data in a space where it results more easily separable. We begin by solving the hyperplane separation problem, and then we describe the mapping to the higher dimensional feature space, where hopefully the data result linearly separable and the optimal hyperplane is obtained by solving the mentioned problem. Further details, proofs and more general results can be found in [57].

Let us first suppose that our data is linearly separable, that is, it exists a hyperplane that separates the two classes. Moreover, let the features of the data $\mathbf{x}_1 \dots \mathbf{x}_m$ belong to a Hilbert space $\mathcal{H}$, and its labels $y_1 \dots y_m \in \{-1, +1\}$.

Let us recall that a hyperplane can be written as:

$$\{\mathbf{x} \in \mathcal{H} \,/ \langle \mathbf{w}, \mathbf{x} \rangle + b = 0\}$$

where $\mathbf{w} \in \mathcal{H}$ is the normal vector and $b \in \mathbb{R}$. Each hyperplane leads to a classification or decision function, namely

$$f_{\mathbf{w},b} : \mathcal{H} \to \{-1, +1\} \tag{4.3}$$

$$f_{\mathbf{w},b}(\mathbf{x}) = sgn\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right) \ , \tag{4.4}$$

Note that we can multiply both $\mathbf{w}$ and b by the same constant, getting the same hyperplane. To avoid this ambiguity we restrict the hyperplanes to the *canonical* ones with respect to the data $x_1 \dots x_m$, meaning that it is satisfied

$$\min_{i=1\dots m} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1 \ , \tag{4.5}$$

so the closest point to the hyperplane has a distance of $1/||\mathbf{w}||$.

Among all hyperplanes separating the data, there exist a unique *optimal hyperplane*, in the sense that it maximize the margin of separation between the data and the hyperplane. Given that the minimum distance of the data to the hyperplane is $1/||\mathbf{w}||$, in order to maximize the margin of separation we have to minimize $||\mathbf{w}||$, keeping a correct classification. This latter constraint can be achieved by asking $y\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right) \geq 0$ for all the training points. However, if that solution indeed exists, the canonicality equation (4.5) implies that $y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\right) \geq 1$. We then end up with the following optimization problem:

$$\min_{\mathbf{w}\in\mathcal{H},b\in\mathbb{R}} \frac{1}{2}||\mathbf{w}||^2 \ , \tag{4.6}$$

$$s.t. \quad y_i\left(\langle \mathbf{w},\mathbf{x}_i\rangle + b\right) \geq 1 \quad \forall i = 1\dots m \tag{4.7}$$

where the constraints (4.7) ensure the correct classification of all the data points and the minimization of the objective function (4.6) ensures the optimality of the hyperplane in terms of the margin of separation.

The Lagrangian associated to this optimization problem is obtained by introducing the Lagrange multipliers $\alpha_i \geq 0$

$$L(\mathbf{w},b,\alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{m} \alpha_i\left(y_i\langle \mathbf{w},\mathbf{x}_i\rangle + b - 1\right) \tag{4.8}$$

From the optimality condition that the derivatives of L must vanish, it follows that

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \ , \tag{4.9}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \ . \tag{4.10}$$

According to the Karush-Kuhn-Tucker theorem (see [57]), given a solution for (4.8), that is, a saddle point $(\bar{\mathbf{w}},\bar{b},\bar{\alpha})$:

$$L(\bar{\mathbf{w}},\bar{b},\alpha) \leq L(\bar{\mathbf{w}},\bar{b},\bar{\alpha}) \leq L(\mathbf{w},b,\bar{\alpha}) \quad , \quad \forall \mathbf{w},b \in \mathcal{H}\times\{-1,+1\}, \alpha\in[0,\infty)^m$$

then the KKT conditions are satisfied:

$$\alpha_i\left(y_i\langle \mathbf{w},\mathbf{x}_i\rangle + b - 1\right) = 0 \quad \forall i = 1\dots m \ .$$

Among all the patterns, those $\mathbf{x}_i$ for which $\alpha_i > 0$ are called *Support Vectors*, and by virtue of the last equation, they lie exactly on the margin. The other patterns are irrelevant in the construction of $\mathbf{w}$ since the corresponding multipliers α_j are zero. Note that this patterns are farther to the hyperplane, so the hyperplane is determined only by its closest patterns (the support vectors), which is reasonable. This is illustrated in Figure 4.11.

Substituting the conditions (4.9) and (4.10) into the Lagrangian (4.8), we obtain the dual formulation of the optimization problem, the quadratic programming problem:

$$\max_{\alpha\in\mathbb{R}} \ \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle \ , \tag{4.11}$$

$$\begin{aligned} s.t. \quad & \alpha_i \geq 0 \quad \forall\, i = 1\dots m \\ & \sum_{i=1}^{m}\alpha_i y_i = 0 \ . \end{aligned}$$

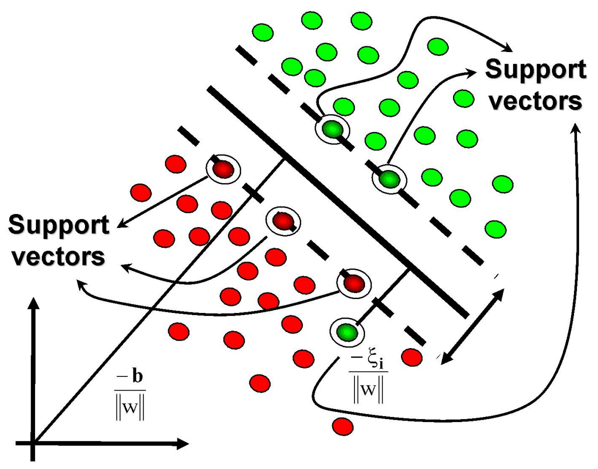

Figure 4.11: Support Vectors and margin.

However, it is reasonable to think that a separating hyperplane may not always exists, and then the constraints (4.7) look very hard. The first idea to relax the problem may be to find the hyperplane that leads to the minimal number of errors, but this problem is NP-hard.

Cortes and Vapnik [17] proposed to relax the problem by introducing new variables $\xi_i \geq 0$, leading to the softer constraints

$$y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\right) \geq 1 - \xi_i \quad \forall i = 1 \dots m \ . \tag{4.12}$$

These constraints can be met for all the instances by making ξ_i large enough, so a penalty term is added to the objective function. In its simplest form, this is known as the C-SV classifier, and the formulation of the optimization problem is as follows:

$$\min_{\mathbf{w}\in\mathcal{H}, b\in\mathbb{R}} \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{m}\sum_{i=1}^{m} \xi_i \ , \tag{4.13}$$

$$s.t. \quad y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\right) \geq 1 - \xi_i \quad \forall i = 1 \dots m \tag{4.14}$$

$$\xi_i \geq 0 \quad \forall i = 1 \dots m$$

In this case, the solution $\mathbf{w}$ has the same form $\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$, and the non-zero coefficients α_i can only occur when the corresponding instance precisely meets the constraint (4.14). The solution of problem (4.13) is found by solving a quadratic programming problem very similar to (4.11).

Unfortunately, the case where the data is (or is close to be) linearly separable is not very frequent. The trick to use the tools developed above is to map the data into a higher dimensional space where, luckily, it results linearly separable. Let us formalize these concepts. The original feature space is now called $\mathcal{X}$, and the higher dimensional feature space is $\mathcal{H}$. In order to use all tools we have just described, $\mathcal{H}$ has to be a Hilbert space. The map Φ is then:

$$\begin{aligned}\Phi &: \mathcal{X} \to \mathcal{H} \\ \Phi(x) &= \mathbf{x} \ .\end{aligned}$$

Very often, the dimension of the resulting feature space $\mathcal{H}$ is so large that it is extremely expensive even to compute $\Phi(x)$. However, if we carefully review the description of the linearly separable case, we can observe that all the computations involve only dot products between elements in $\mathcal{H}$, and never vectors in $\mathcal{H}$ by themselves. Therefore, taking into account this last observation, we can design the map Φ in such a way that the computation of dot products $\langle \Phi(x), \Phi(x') \rangle$ results computationally affordable. More specifically, we employ representations Φ such that the dot product $\langle \Phi(x), \Phi(x') \rangle$ corresponds to a *kernel* k:

$$\langle \Phi(x), \Phi(x') \rangle = k(x, x')$$

which allow us to compute the dot product without having to explicitly compute the map Φ.

Let us present some examples of the most used kernels, starting with the polynomial kernel. Here Φ maps each $x \in \mathbb{R}^n$ to the vector $\mathbf{x} = \Phi(x)$ whose entries are all the possible d-th degree products of the entries of x, scaled by the number of number of occurrences of each monomial. For example, for $n = 2$ and $d = 2$ we get:

$$\begin{aligned}\Phi &: \mathbb{R}^2 \to \mathcal{H} = \mathbb{R}^3 \\ \Phi(x_1, x_2) &= (x_1^2, x_2^2, \sqrt{2}x_1x_2)\end{aligned}$$

It is easy to see that for the polynomial kernels, it holds

$$\langle \Phi(x), \Phi(x') \rangle = \langle x, x' \rangle^d \ .$$

The second example family is the one of the Radial Basis Function (RBF) kernels, which are those that can be written in terms of a metric $d(x, x')$ on $\mathcal{X}$, as

$$k(x, x') = f(d(x, x')) \ .$$

For example, the Gaussian kernel is

$$k(x, x') = e^{-\frac{||x-x'||^2}{2\sigma^2}} \ ,$$

and it is, with the polynomial kernel, one of the classic kernels used in pattern recognition.

Note that the map Φ is not needed for the computations, and therefore it does not care if we ignore the explicit form of Φ. In the case of the Gaussian RBF kernel, the feature space $\mathcal{H}$ is infinite-dimensional.

Figure 4.12 shows an example of how the embedding of the data on a higher dimensional space can simplify the desicion boundary. Given two classes, assume that

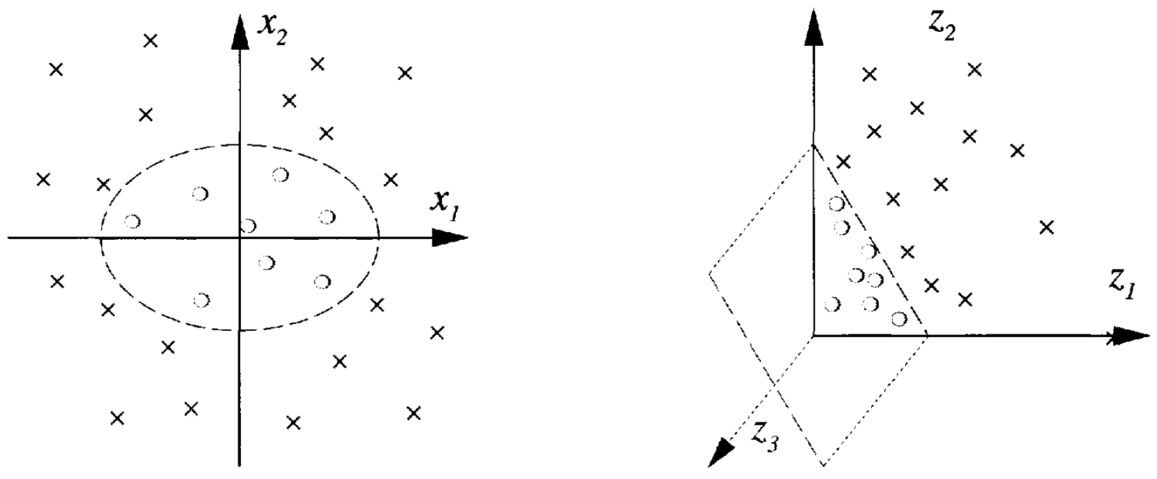

Figure 4.12: Dataset mapped by polynomial kernel. Figure from [57].

the true optimal decision boundary is an ellipse. Then, under the mapping $\Phi(x_1, x_2) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$, this ellipse becomes a plane which, of course, separates the classes.

The classifying technique C-SVM, intensively used in this work, is obtained by applying this kernel trick to the hyperplane separation approach, specifically to the problem C-SV in equation (4.13). The chosen kernel for the feature space change is the polynomial kernel. An optimal choice of the parameters (C, d) is performed using grid-search. More details about the parameters are provided in the next chapter.

4.3.2 Classification Trees + AdaBoost

A classification tree is a model that classifies a new instance based on the values of the features, following a tree structure (Figure 4.13). Given the tree and the new instance, starting from the root, at each node a decision is made according to the value of a feature, descending trough the nodes until reaching a leaf. The leafs are labeled, and the label of the reached leaf is the class predicted by the classifier. This classification process, once the tree is constructed, is a simple sequence of decisions based on the features values, so the model is very intuitive.

Given a training set, the problem of finding the optimal decision tree is NP-Complete, so heuristic criteria are used to construct the tree. For example, the *C4.5* [53] is an algorithm used to construct desicion trees that starts with a node containing the complete training set, and it splits the node according to the feature that better separates the data, in the sense that the new nodes are as pure[4] as possible. These criteria to choose the better feature at each node is based on measuring the impurity with the Information Gain (or Kullback-Leibler divergence), and keeping the feature that maximizes the

[4]A node is said to be *pure* if all the instances belong to the same class.

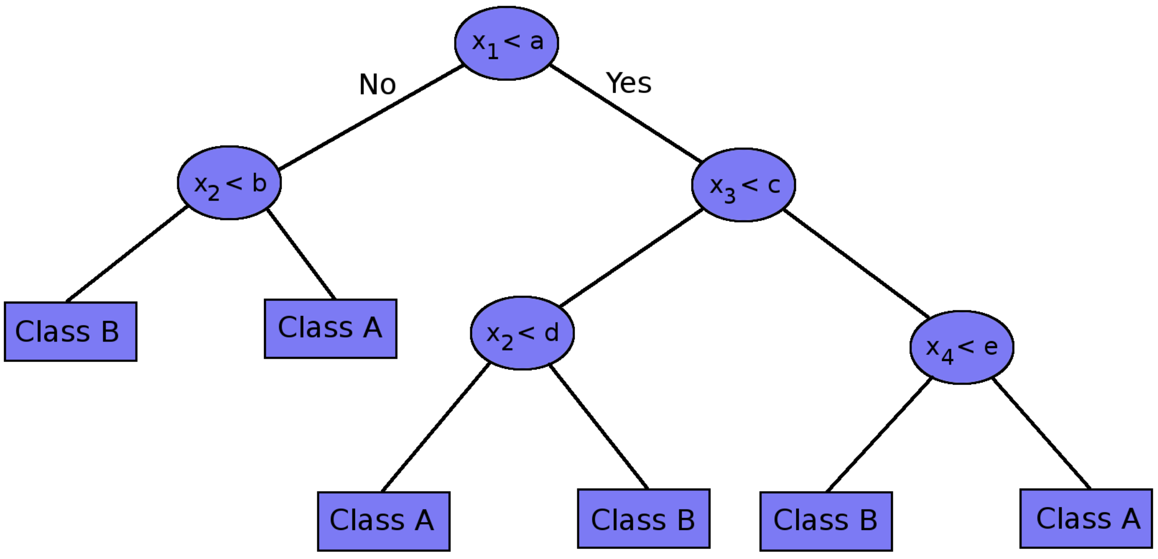

Figure 4.13: Scheme of typical classification tree.

impurity decrease. More specifically, assuming only two classes w_1 and w_2, at each node N the information impurity can be measured as

$$i(N) = -\sum_{j=1}^{2} P(w_j) \log_2 P(w_j)$$

where $P(w_j) \in [0,1]$ is the proportion of instances of the w_j class at node N. This is the binary entropy of the distribution $(P(w_1), P(w_2))$, so it is minimum (zero) when all the instances belong to a single class, it reaches its maximum (one) when the instances are equally balanced, and it is symmetric. See Figure 4.14. Therefore, this is a measure of how pure is the node.

Now, the feature to pick at node N is the one that maximize the impurity decrease:

$$\Delta i(N) = i(N) - P_L i(N_L) - (1 - P_L) i(N_R)$$

where N_L and N_R are the nodes obtained by splitting the node N with the testing feature, and P_L is the proportion of instances in the left node N_L.

Usually, the desicion trees tend to overfit the training data (and therefore they do not generalize well to new data), so a pruning technique is often used to reduce this effect.

These trees are generally unstable, in the sense that they can produce significant different trees from slightly different training datasets. On the one hand, this is a drawback from the robustness point of view, but on the other hand, some techniques that take classifiers as input, require precisely this instability. This is the case, for example, of the MetaCost algorithm used for the imbalance problem, or the AdaBoost, commented below.

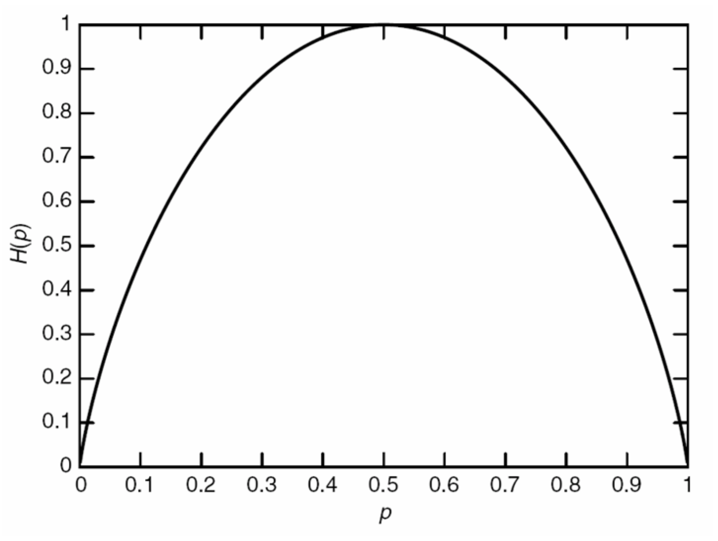

Figure 4.14: Entropy H of the distribution $(p, 1-p)$.

The AdaBoost (Adaptive Boosting) [28] is a technique to combine classifiers. One may ask why is it convenient to combine multiple classifiers instead of choosing the best classifier available. At least three reasons can be given in defense of the classifier combination approach [39, 18].

- The first one is statistical. Suppose that we have several classifiers with good performance over the training data set. These different classifiers may have very different generalization properties, so by choosing one of them we may have a considerable risk of picking a classifier with bad performance over other real data. A more conservative option is to combine all of them, so this new classifier may not be as good as the better available classifier, but we will reduce the risk of choosing a bad classifier.

- The second reason is computational. Since some training algorithms follow a greedy-like search, they may end up in different local optima. The combination of these classifiers may lead to a better approximation of the best classifier.

- And finally the third reason is representational. It may be the case that the best classifier does not lie in the classifier space where we are looking for (for example, linear classifiers or trees). However, a combination of these classifiers may reach a performance similar to the performance of the best classifier (for instance, we can approximate any decision boundary with an ensemble of linear classifiers).

A simple numerical example throws some light on the practical side. If we have 9 independent classifiers with probability 0.8 of giving the correct class, then the classifier obtained by majority vote over the 9 classifiers have an accuracy of 0.98.

Back to the specific case of AdaBoost, the idea in this case is to train the classifiers carefully and incrementally, and then combine them linearly. More specifically, given a training dataset $\mathbf{x}_1 \dots \mathbf{x}_m$ with labels $y_1 \dots y_m$, the first classifier is trained with a subset of the dataset, sampled using a uniform distribution. The second classifier is trained with another sample of the same training data but with an updated distribution that assigns more likelihood to those instances that where incorrectly classified by the first classifier. This way, the second classifier will pay more attention to the "difficult" instances in the dataset. This process continue, updating the distribution taking into account the classification errors of the previous step and training the new classifier with the dataset sampled under this distribution, until training all the T classifiers. The final classifier is

$$C(\mathbf{x}) = sign\left(\sum_{k=1}^{T} \alpha_k D_k(\mathbf{x})\right)$$

where $D_k(\mathbf{x}) \in \{-1, +1\}$ is the label assigned by the classifier k, and α_k is a weight factor computed at each step (along with the new dataset distribution), depending on the classification error of the classifier k. The algorithm in Figure 4.15 is a pseudo-code of the AdaBoost, more details of this algorithm can be found in [28, 39].

```
Given $(x_1, y_1) \dots (x_m, y_m) \in \mathbb{R}^n \times \{-1, +1\}$;
Set the weights $w_i^1 = \frac{1}{m}$;
**for** $k$=1 **to** $T$ **do**
    Take a sample $S_k$ from de dataset using distribution $\mathbf{w}^k$;
    Train classifier $D_k$ using $S_k$;
    Compute weighted error at step $k$:
        $$\varepsilon_k = \sum_{i=1}^{m} w_i^k l_{k,i}$$
    where $l_{k,i} = 1$ if $D_k$ misclassifies the instance $(x_i, y_i)$ and 0 otherwise Compute
        $$\beta_k = \frac{\varepsilon_k}{1 - \varepsilon_k}$$
    Update the weights
        $$w_j^{k+1} = \frac{w_j^k \beta_k^{1-l_{k,i}}}{\sum_{i=1}^{m} w_i^k \beta_k^{1-l_{k,i}}}$$
**end**
Return $D_1 \dots D_T$
```

Figure 4.15: Training phase of AdaBoost.

4.3.3 Other tested classifiers

Of course, we tested several classifiers in order to choose the best solution to our problem. Among these classifiers, we will point out two of them.

The first one is the Naïve Bayes classifier, which is a very simple classifier. The Naïve term is due to the fact that this classifier assumes independence of the features $(x_1, \ldots, x_n)$. Under this assumption, is then easy to estimate the distributions $p(\mathbf{x}|y)$ as $p(\mathbf{x}|y) = \prod_{i=1}^{n} p(x_i|y)$. The estimation of $p(\mathbf{x}|y)$ is the basis of the Bayesian classifiers. Given a new instance $\mathbf{x}$ the decision is taken as the *maximum a posteriori* (MAP), this is

$$\arg \max_{\text{class c}} p(y = c) p(\mathbf{x}|y = c) \ .$$

This classifier is then very simple, it can easily work with a large number of features and its performance may be good enough even when the independence assumption is violated. Indeed, when we tested this classifier with our dataset, the performance was close to (but not as good as) the obtained with SVM.

The second classifier in this section, the One-Class SVM, was tested as an alternative solution to the class imbalance problem, in the following sense. Instead of considering two classes with a great imbalance and trying to apply a classical binary classifier with some previous step to balance the data, the idea is to consider only one class, whose data is enough to train the classifier, and then classify the new instances as belonging to the chosen class if the feature match with the model, or as outliers if they do not. In the case of the application in this work, the polyps would be the outliers, and the non-polyps the class for which the model is trained.

Chapter 5

Results and Conclusions

In this chapter we give details on the parameters selection, evaluation process, database used, some implementation details, etc. Then we present the obtained numerical results and compare them with the state-of-the-art results presented in the Section 1.5. Finally, we give a qualitative discussion about the results, future work and conclusions.

5.1 Classification framework and details

5.1.1 Database used

A total of 31 cases of the WRAMC database [52] were used to test the proposed CAD algorithm,[1] with 49 polyps detected by optical colonoscopy, including two flat polyps. Among these 49 polyps, 34 are larger than $6mm$ in size, and the other 15 are between $3mm$ and $6mm$ in size. Figure 5.1 shows the histogram of the polyps' size.

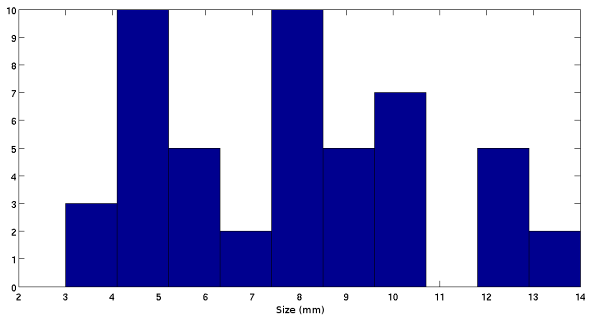

Figure 5.1: Histogram of the polyps sizes.

This database contains around 800 studies (not all of them containing polyps), but the information about the polyps location is very poor, and it is very hard to get a precise location. We got access to some documents describing with more detail the polyps information, so we could identify these polyps in our segmentation and determine

[1]Data provided courtesy of Dr. Richard Choi, Virtual Colonoscopy Center, Walter Reed Army Medical Center.

their precise coordinates. Unfortunately we have only a few of these documents, and they are no longer available on the web.

Patient preparation and other details about the study can be found in [52]. In particular, the preparation process was standard, with oral administration of 90ml of sodium phosphate and 10mg of bisacodyl (laxatives). Additionally, as part of the clear-liquid diet, the patients consumed 500ml of barium for solid-stool tagging and 120ml of diatrizoate meglumine and diatrizoate sodium for the opacification of luminal fluid. A 4-channel or 8-channel CT scanner was used (GE LightSpeed or LightSpeed Ultra, General Electric Medical Systems).

5.1.2 Implementation details

All the coding was done in *C++* and compiled with *gcc* under GNU/Linux. The *ITK* [33] libraries were used as basis to implement the majority of the image processing algorithms, and the *VTK* [58] libraries were used for visualization and also for some filter coding.

Among others, we implemented the Geodesic Active Regions algorithm, all the surface motions by curvature, and the whole preprocessing stage in the segmentation (the probability map computation). We also implemented all the code of the Chapter 4, with the only exception of the geodesic distance computation, which was kindly provided by Alberto Bartesaghi [5].

The software package *Weka* [30] was used for the classification and feature selection. In particular we used the implementation of the following algorithms: SVM, C4.5 trees, Naïve Bayes, AdaBoost, CostSensitive learning, MetaCost and SMOTE,

5.1.3 Evaluation scheme and parameters selection

As commented in Chapter 4, in order to make a fair evaluation of the system the instances used for training should not be used for testing. A common scheme used in virtual colonoscopy works, very similar to the n-folds cross validation, is the so-called "leave-one-out" cross validation, which in this case means training with 30 studies and testing with the remaining one.

The spirit of the evaluation was to achieve 100% sensitivity (if possible), with the lowest false positive rate possible. Toward this aim, for each algorithm (classifiers and methods for dealing with class imbalance problem) the parameters were chosen testing all the possible combinations in a grid, and keeping the set that reaches the highest sensitivity. At the same sensitivity, the parameters with the lowest false positive number were preferred.

The number of iterations for the smoothing step (Section 3.3) can be set by choosing the value that maximizes the overall performance of the system, as we have just described. Alternatively, we can consider a sphere of the size of the resolution and compute analytically the number of iterations that are needed to make it vanish. The idea behind this procedure is to smooth the surface up to the resolution limit. These two approaches led to the same result, 15 iterations, and therefore this is the chosen value for the parameter. The other parameter involved in the segmentation step is the iso-level

α for the marching cubes algorithm, whose value does not influence the segmentation result as illustred in Figure 3.15.

After the grid-search over the three remaining parameters ((C, d) of SVM and the cost of CostSensitive learning), we end up using CostSensitive learning with a cost matrix:

$$\begin{pmatrix} 0 & 35 \\ 1 & 0 \end{pmatrix}$$

and C-SVM with $C = 1$ and Polynomial kernel with exponent 1.5.

5.2 Numerical results

The numerical results listed below were obtained by classifying with SVM using Cost Sensitive learning with the best parameters (in the sense of the previous section), after normalizing the data; Naïve Bayes performed similarly.

These values are comparable with the state-of-the-art results [73, 66], but our database includes very small and therefore challenging polyps. A more precise comparison of results is not necessarily useful, since in general each work considers its own database.

When testing with all the polyps, a 100% sensitivity is achieved with an average of 2.2 false positives FPs per patient case. On the other hand, when testing our CAD pipeline with the polyps greater than $6mm$ in size (34 in total, including two flat polyps), which is the typical framework in the VC CAD literature, the results are further improved, a 100% sensitivity is achieved with just 0.8 FPs per study. Figure 5.2 shows the FROC curve obtained using the whole database (polyps $> 3mm$ in size) and Table 5.1 compares the classification results according to the polyps size. Again, the work with such small, as well as flat polyps, is unique to the framework here presented; see next.

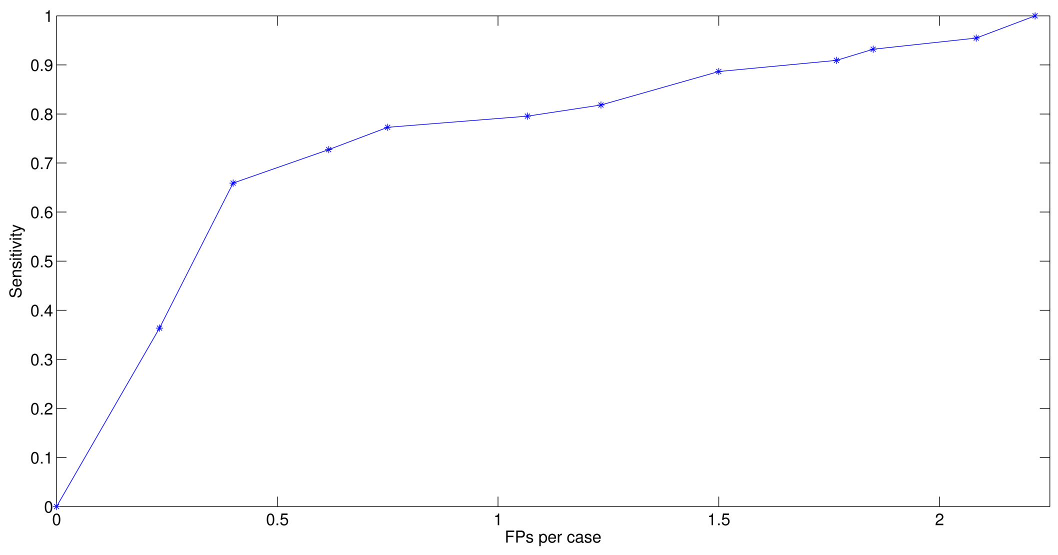

Figure 5.2: FROC curve for the proposed system.

	Polyp sizes	
	$> 3mm$	$> 6mm$
Sensitivity	100%	100%
FP per case	2.2	0.8

Table 5.1: Comparison of performance by polyp size.

Differential features importance

Table 5.2 shows the comparison between absolute and differential texture features. The classification was performed using all the geometric features and either the absolute texture features (computed just for V_1), or the differential texture features, using the standard leave-one-out strategy. The results show that, when combined with the differential geometric features, differential texture features are significantly more discriminative than the absolute ones. The FROC curve in Figure 5.3 extends the results in Table 5.2 and compares the performance of the classifier when using differential or absolute features.

	Texture features	
	Absolute	Differential
Sensitivity	96%	100%
FP per case	3.1	2.2

Table 5.2: Comparison of absolute and differential texture features, with polyps greater than $3mm$ in size.

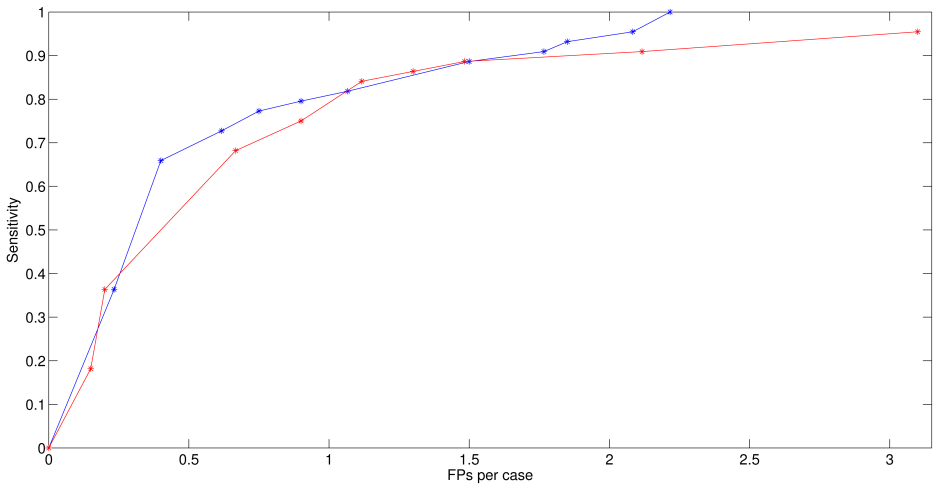

Figure 5.3: FROC curve comparing the performances with differential (in blue) and absolute (in red) texture features.

Segmentation comparison

To compare objectively our segmentation method with the previously provided segmentation, we tested the results over the same dataset (10 cases). The best classification results for each segmentation method are shown in Table 5.3, where it can be seen the significant improvement of our segmentation scheme in terms of polyp detection.

	Segmentation	
	Provided	Our method
Sensitivity	100%	100%
FP per case	6.6	0.9

Table 5.3: Comparison of performance using different segmentations, for polyps greater than $6mm$ in size.

On the other hand, the FROC curve in Figure 5.4 compares the performance of the system with the different smoothing methods discussed in Chapter 3. It can be seen that the proposed smoothing technique achieves better results than the others.

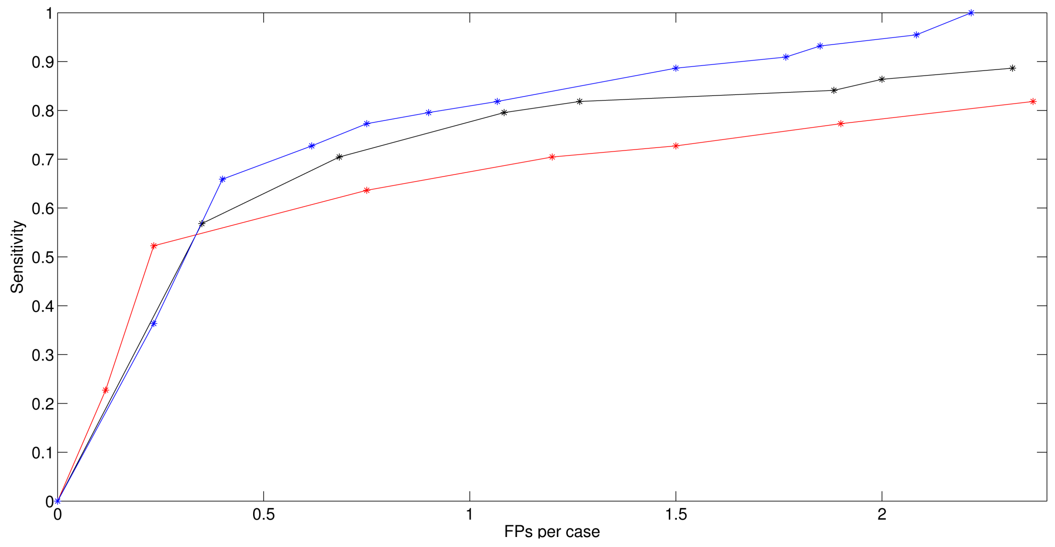

Figure 5.4: FROC curve comparing the performances using the different smoothing methods. The curve for the proposed evolution is shown in blue, and the results for the evolution by $\mathcal{H}$ and κ_{min} are shown in red and black respectively.

Finally, Table 5.4 is the same that was presented when the prior work was discussed (Chapter 1, Section 1.5), but with our results added. Once again, different databases were used in different works so these comparisons might not be completely fair, but this table shows that our algorithm is the only one capable to detect medium-size polyps ($> 6mm$) with less than one false positive per study. In addition, our system detect small polyps ($> 3mm$) with the lowest false positive rate.

Technique	# Polyps	Size	Sensitivity	FP per study
Summers *et al.* [64]	≈ 30	$> 10mm$	89.3%	2.1
Summers *et al.* [64]	≈ 120	$> 6mm$	60%	8
Yoshida *et al.* [76]	12	$> 5mm$	100%	2
Paik *et al.* [49]	7	$> 10mm$	100%	7
Wang *et al.* [73]	61	$> 4mm$	100%	2.68
Hong *et al.* [32]	≈ 120	$> 5mm$	100%	3
Sundaram *et al.* [65]	20	$> 10mm$	100%	18
Sundaram *et al.* [65]	122	$> 2mm$	80%	24
van Wijk *et al.* [69]	57	$> 6mm$	95%	5
van Wijk *et al.* [69]	32	$> 10mm$	95%	4
Suzuki *et al.* [66]	28	$> 5mm$	100%	1.1
Bogoni *et al.* [7]	21	$> 5mm$	90%	> 3
Taylor *et al.* [67]	32	$> 6mm$	81%	13
Proposed pipeline	34	$> 6mm$	100%	0.8
Proposed pipeline	49	$> 3mm$	100%	2.2

Table 5.4: Performance comparison of our method with the results reviewed in Chapter 1.

5.3 Discussion

5.3.1 Small polyps, big polyps, flat lesions

It is clear that both the small polyps and flat lesions are much more difficult to detect than the other polyps. What is not clear is if the same kind of algorithm and features are suitable for detecting all the range of polyp types and sizes. We showed that the proposed combination of features, although it may not be optimal for every specific type of lesion, is able to correctly detect all of them.

All the stages in the pipeline, specially the segmentation and the features, contribute to the good classification results for the whole database. However, it would be interesting to study which pre-processing techniques and features are better for each type and size of polyps, and eventually propose different CAD systems for each class of polyp. Nevertheless, the 100% sensitivity together with the 2.2 FP rate for polyps greater than $3mm$ in size is as remarkable as the 0.8 FP rate for polyps beyond $6mm$ in size.

Some examples of the classified polyps are shown in Figure 5.5.

5.3.2 Geometric and texture importance

Although the geometric features are the most discriminative ones, the texture features still play a fundamental role in the classification. Adding the texture features to the geometric ones, the sensitivity reaches 100%, and at the same time the false positives rate decreases by 30%.

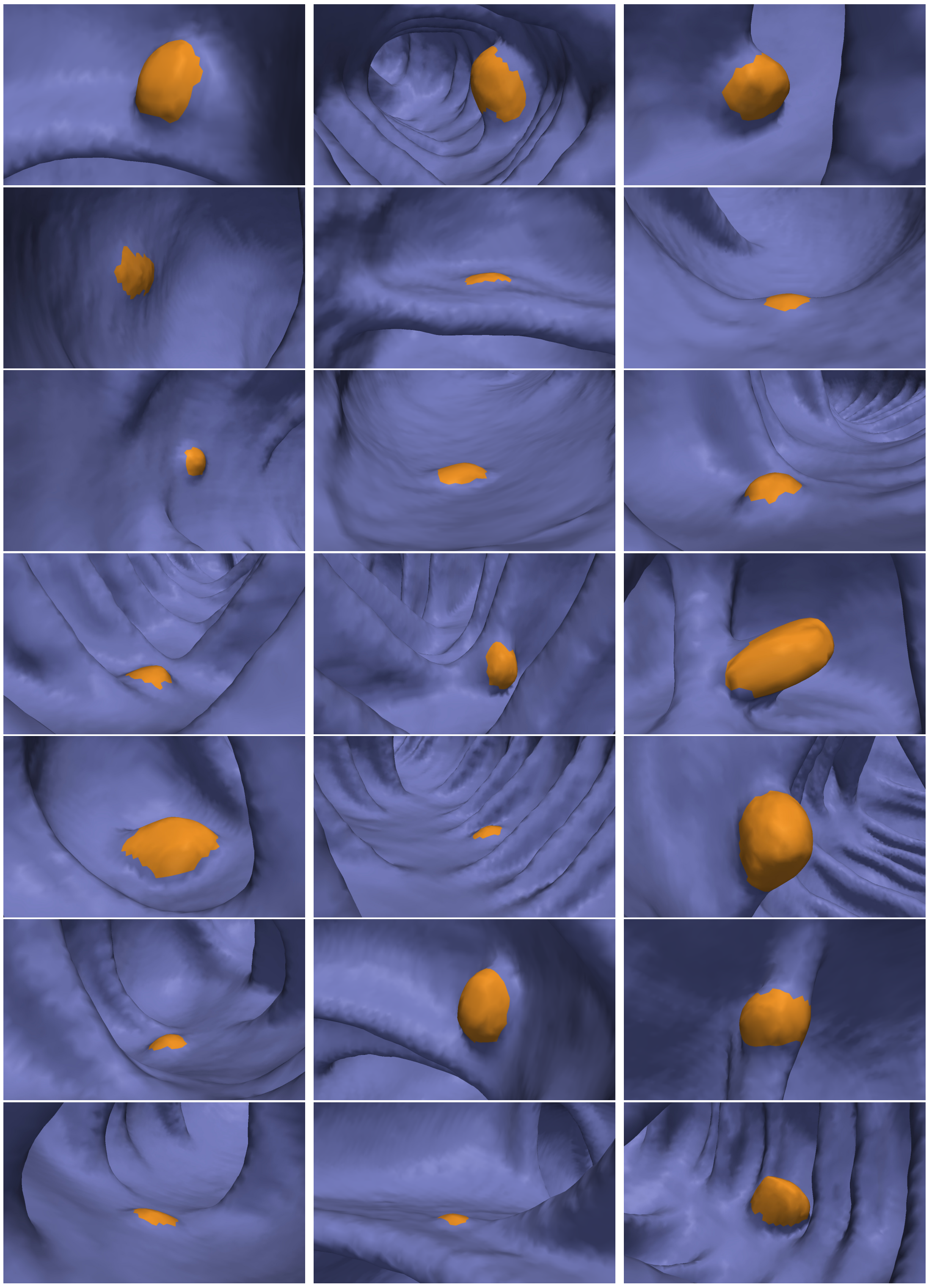

Figure 5.5: Example of patches corresponding to well-classified polyps..

Figure 5.6 shows a detected polyp, where the geometry is crucial, because the gray-level does not present considerable local variations. This is specially true for polyps located over tagged material. On the other hand, for the flat polyp in Figure 5.7, the geometry is weakly discriminative (although the measure considering the ring enhances the detectability), and the texture features lead to a correct classification.

Texture information is very important also because it is more robust to segmentation errors, as the texture features are computed integrating from the volumetric data itself (once the local volumes have been considered). Moreover, the differential texture features (the differences between **P** and **R**), outperform the absolute texture features (just computed in **P**), as shown in Table 5.2.

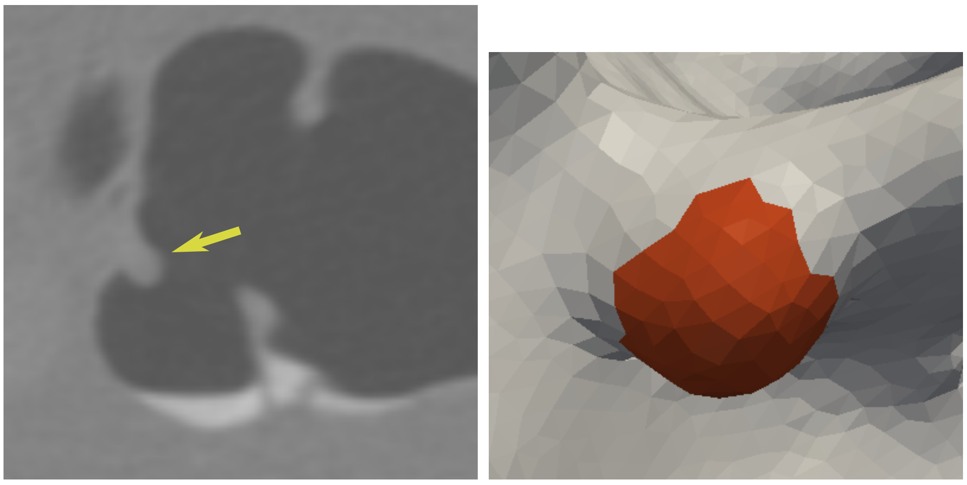
Figure 5.6: Polyp with no texture information.

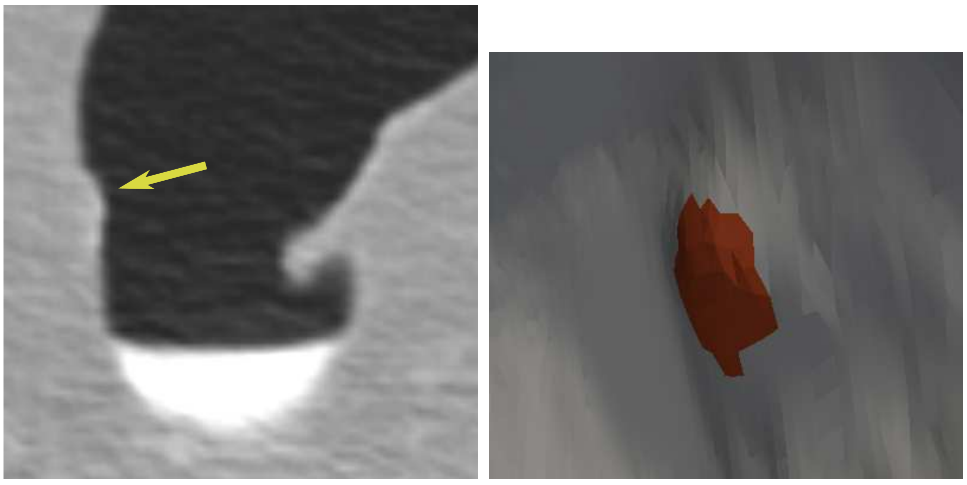
Figure 5.7: Polyp with texture information, but weak geometric information.

5.3.3 Qualitative analysis of false positives

In addition to the number of false positives, it is very important to study how these FP patches look like, since some of them can be quickly ruled out by the expert and some can be avoided by improving some aspects of the segmentation step.

About half of the false positives are quite reasonable, in the sense that they are (usually small) sections of the colon that are polyp-like shaped, Figure 5.8. This is specially reasonable taking into account that we designed the system to also detect small and flat polyps. On the other hand, about 20% of the FPs were in fold sections of the wall (Figure 5.8) or on parts of the insufflation tube. All these patches are easily ruled out by inspection. Another 20% of the FPs were caused by segmentation errors, this number being significantly reduced thanks to the texture features as discussed above.

More examples of false positives are shown in Figure 5.9. There are two false positives caused by segmentation errors. However, Figure 5.10 shows two false positives that might look like segmentation errors, but the CT images show that the segmentation is reasonable. The reason of these false positives may be fecal residues, or they might be true polyps missed by the optical colonoscopy.

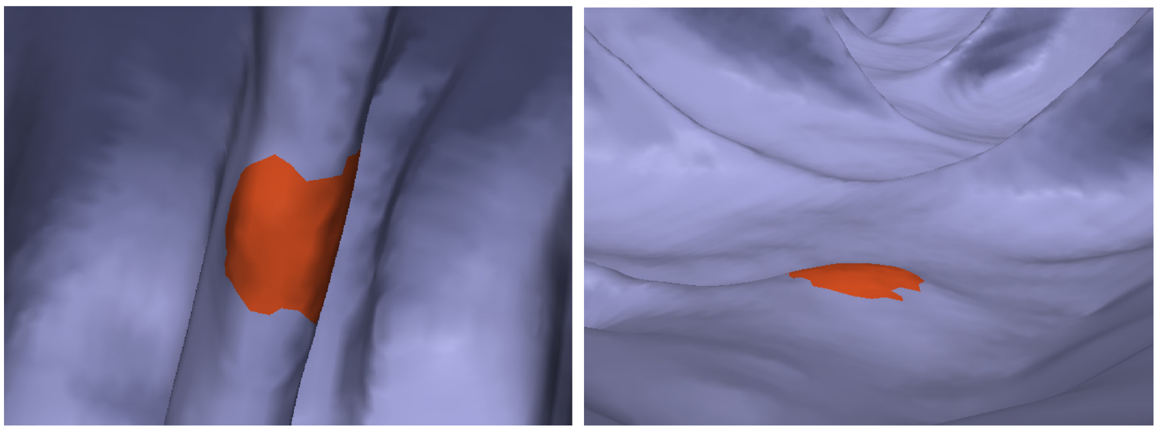
Figure 5.8: False positives: fold and patch similar to polyp.

5.4 Future work

There are not few directions to continue improving this work. In terms of the scope, this thesis is focused on CT studies with a certain bowel preparation. It is well known that the preparation is the most uncomfortable step for the vast majority of patients, and therefore it is interesting to study how do the CAD algorithms perform with a different bowel preparation, less favorable for the segmentation.

Even with the preparation process here presented, the segmentation may be improved. We think that the major difference could be in the preprocessing stage, given that the smoothing step seems to perform as good as it can.

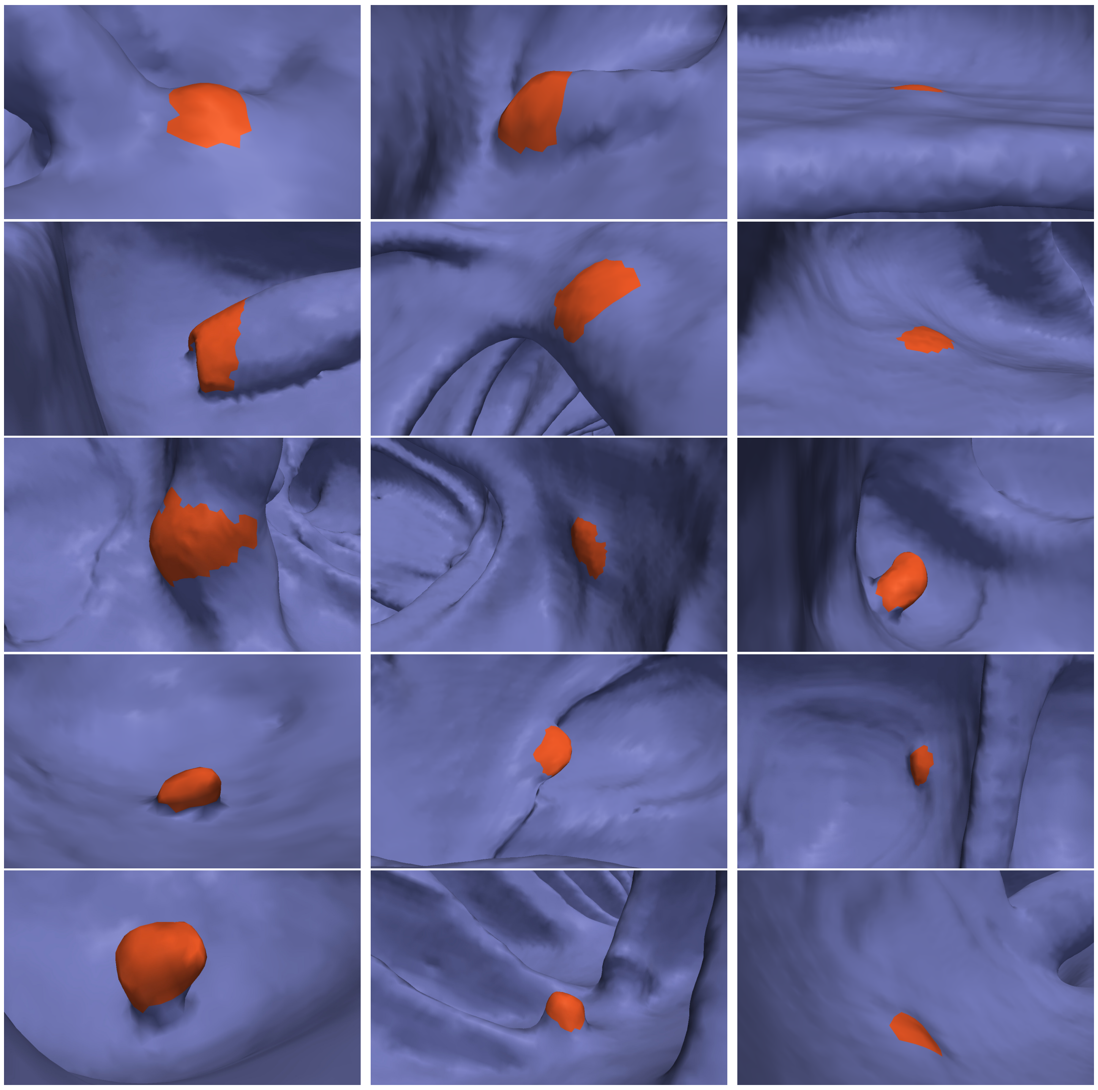

Figure 5.9: Examples of false positives.

The inclusion of new features may help the classification step. On the one hand, there might be better measures for the characteristics that we want to describe. On the other hand, specialized doctors may have other ideas about what to look for when searching for polyps, and these ideas may lead to new features.

It would be very helpful to count with a larger database, with a proper labeling of the polyps. More and better experiments would be possible, new problems and new ideas would arise for sure.

Finally, the histologic information of the polyps is very important. With a large database containing the histological pattern of each polyp (obtained from biopsy), it might be possible to propose features and train machine learning algorithm to classify the polyps according to its histology.

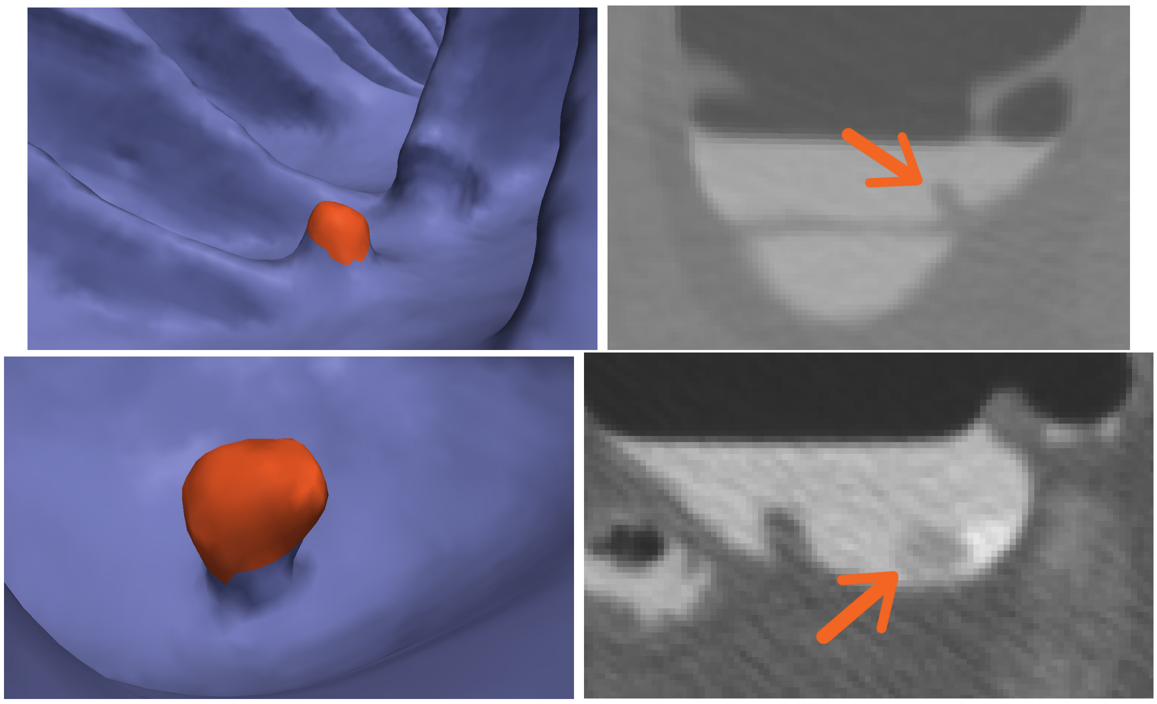

Figure 5.10: False positives patches with their corresponding CT slice.

5.5 Conclusions

We introduced a complete pipeline for a Computer-Aided Detection algorithm that flags candidate polyp regions. This CAD algorithm is completely automatic. All the stages of the system were implemented producing state-of-the-art results.

The segmentation stage is simple and computationally efficient, it includes several novelties and it produces high quality segmentations. Maybe the main novelty is the smoothing PDE which enhances the polyps, leading to a better detection.

The new multi-scale approach for delineating the candidate patches allows to precisely delineate polyps.

The main novelties of the classification stage are in the consideration of the surrounding area for each candidate polyp (we compute differential features instead of absolute ones), and the inclusion of the Haralick texture features.

As a consequence, the system is able to detect polyps greater than $3mm$ in size with only 2.2 false positives per case. When looking for polyps greater than $6mm$ in size, our system can achieve 100% sensitivity with less than one false positive per case. Although tested with different datasets, these results are within the best in the field.

Bibliography

[1] L. Alvarez, F. Guichard, P. L. Lions, and J.-M. Morel. Axioms and Fundamental Equations of Image Processing. 123:199–257, 1993.

[2] American Cancer Society. Colorectal cancer facts & figures 2008-2010, http://www.cancer.org/research/cancerfactsfigures, 2010.

[3] A. J. Aschoff, A. S. Ernst, H.-J. Brambs, and M. S. Juchems. CT colonography: an update. *European Radiology*, 18(3):429–37, Mar. 2008.

[4] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations (Applied Mathematical Sciences)*. Springer, 2nd edition, August 2006.

[5] A. Bartesaghi. Generation of weighted geodesic curves over triangulated domains. Master's thesis, Facultad de Ingeniería, Universidad de la República, 2000.

[6] A. Blachar, M. Graif, A. Kessler, and J. Sosna. State-of-the-art CT colonography: Update on technique and performance. *Current Colorectal Cancer Reports*, 3(1):49–54, May 2007.

[7] L. Bogoni, P. Cathier, M. Dundar, A. Jerebko, S. Lakare, J. Liang, S. Periaswamy, M. E. Baker, and M. Macari. Computer-aided detection (CAD) for CT colonography: a tool to address a growing need. *Br J Radiol*, 78(suppl_1):S57–62, 2005.

[8] J. H. Bond. Colorectal cancer screening: the potential role of virtual colonoscopy. *Journal of Gastroenterology*, pages 92–96, 2002.

[9] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic Active Contours. *International Journal of Computer Vision*, 22(1):61–79, February 1997.

[10] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. Minimal surfaces: a geometric three dimensional segmentation approach. *Numerische Mathematik*, pages 423–451, 1997.

[11] V. Caselles and C. Sbert. What is the best causal scale space for three-dimensional images? *SIAM Journal on Applied Mathematics*, 56(4):1199–1246, 1996.

[12] M. E. Celebi, H. Kingravi, B. Uddin, H. Iyatomi, Y. A. Aslandogan, W. Stoecker, and R. Moss. A methodological approach to the classification of dermoscopy images. *Comput Med Imaging Graph*, 31(6):362–73, 2007.

[13] T. Chan and L. A. Vese. Active contours without edges. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 10(2):266–77, Jan. 2001.

[14] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[15] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[16] D. Chopp and J. Sethian. Flow under curvature: singularity formation, minimal surfaces, and geodesics. *Experimental Mathematics*, 2(4):235–255, 1993.

[17] C. Cortes. Support-vector networks. *Machine learning*, 20(3):273–297, Sept. 1995.

[18] T. Dietterich. Ensemble methods in machine learning. *Multiple classifier systems*, pages 1–15, 2000.

[19] M. Do Carmo. *Differential geometry of curves and surfaces*, volume 3. Prentice-Hall Englewood Cliffs, NJ, 1976.

[20] P. Domingos. MetaCost: A General Method for Making Classifiers Cost-Sensitive. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 155–164, 1999.

[21] R. Duda, P. Hart, and D. Stork. *Pattern classification*, volume 2. Citeseer, 2001.

[22] C. Elkan. The foundations of cost-sensitive learning. In *In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 973–978, 2001.

[23] J. Fidler and C. Johnson. Flat polyps of the colon: accuracy of detection by CT colonography and histologic significance. *Abdom Imaging*, 34(2):157–71, 2009.

[24] J. L. Fidler, C. D. Johnson, R. L. MacCarty, T. J. Welch, A. K. Hara, and W. S. Harmsen. Detection of flat lesions in the colon with CT colonography. *Abdom Imaging*, 27(3):292–300, 2002.

[25] M. Fiori, P. Musé, S. Aguirre, and G. Sapiro. Automatic colon polyp flagging via geometric and texture features. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, volume 1, pages 3170–3173. IEEE, Jan. 2010.

[26] M. Fiori, P. Musé, and G. Sapiro. A complete system for polyps flagging in virtual colonoscopy. *IMA Preprint Series*, 2367, 2011.

[27] W. Firey. Shapes of worn stones. *Mathematika*, 21:1–11, 1974.

[28] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, volume 139, pages 23–37. Springer, 1995.

[29] S. B. Götkürk, C. Tomasi, B. Acar, C. F. Beaulieu, D. S. Paik, R. B. Jeffrey, J. Yee, and S. Napel. A statistical 3-D pattern processing method for computer-aided detection of polyps in CT colonography. *IEEE Trans Med Imaging*, 20(12):1251–60, 2001.

[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The WEKA data mining software: an update. *Special Interest Group on Knowledge Discovery and Data Mining Explorer Newsletter*, 11(1):10–18, Nov. 2009.

[31] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, 3(6):610–621, 1973.

[32] W. Hong, F. Qiu, and A. Kaufman. A pipeline for Computer Aided Polyp Detection. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):861–868, 2006.

[33] L. Ibanez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-15-7, http://www.itk.org/ItkSoftwareGuide.pdf, second edition, 2005.

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

[35] D. Kim, P. Pickhardt, A. Taylor, and W. Leung. CT colonography versus colonoscopy for the detection of advanced neoplasia. *New England Journal of Medicine*, 358(1):89; author reply 90, Jan. 2007.

[36] R. Kimmel and J. Sethian. Computing geodesic paths on manifolds. In *Proc. Natl. Acad. Sci. USA*, pages 8431–8435, 1998.

[37] J. J. Koenderink. *Solid shape*. MIT Press, Cambridge, USA, 1990.

[38] E. Konukoglu, B. Acar, D. S. Paik, C. F. Beaulieu, J. Rosenberg, and S. Napel. Polyp enhancing level set evolution of colon wall: method and pilot study. *IEEE Trans Med Imaging*, 26(12):1649–56, Dec. 2007.

[39] L. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.

[40] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169. ACM, 1987.

[41] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM.

[42] E. Meinhardt, E. Zacur, A. F. Frangi, and V. Caselles. 3D Edge Detection by Selection of Level Surface Patches. *Journal of Mathematical Imaging and Vision*, 34(1):1–16, Oct. 2008.

[43] O. Monga, S. Benayoun, and O. Faugeras. From partial derivatives of 3-D density images to ridge lines. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 354–359, 1992.

[44] J.-M. Morel, F. Guichard, and R. Ryan. *Contrast invariant image analysis and PDE's.*

[45] J. Näppi and H. Yoshida. Automated detection of polyps with CT colonography: evaluation of volumetric features for reduction of false-positive findings. *Academic radiology*, 9(4):386–97, Apr. 2002.

[46] P. J. Olver, G. Sapiro, and A. Tannenbaum. Invariant Geometric Evolutions of Surfaces and Volumetric Smoothing. *SIAM Journal on Applied Mathematics*, 57(1):176–194, 1997.

[47] S. Osher, R. Fedkiw, and K. Piechor. *Level Set Methods and Dynamic Implicit Surfaces*, volume 57. 2004.

[48] S. Osher and J. A. Sethian. Fronts Propagating with Curvature- Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[49] D. S. Paik, C. F. Beaulieu, G. D. Rubin, B. Acar, R. B. Jeffrey, J. Yee, J. Dey, and S. Napel. Surface normal overlap: a computer-aided detection algorithm with application to colonic polyps and lung nodules in helical CT. *IEEE Trans Med Imaging*, 23(6):661–75, 2004.

[50] N. Paragios and R. Deriche. Geodesic active regions for motion estimation and tracking. In *ICCV (1)*, pages 688–694, 1999.

[51] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision*, pages 223–247, Feb. 2002.

[52] P. Pickhardt, J. Choi, I. Hwang, J. Butler, M. Puckett, H. Hildebrandt, R. Wong, P. Nugent, P. Mysliwiec, and W. Schindler. Computed tomographic virtual colonoscopy to screen for colorectal neoplasia in asymptomatic adults. *New England Journal of Medicine*, 349(23):2191–2200, 2003.

[53] R. J. Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[54] G. Sapiro. *Geometric Partial Differential Equations and Image Analysis.* Cambridge University Press, 2001.

[55] G. Sapiro and A. Tannenbaum. Affine Invariant Scale-Space. *International Journal of Computer Vision*, 44, 1993.

[56] G. Sapiro and A. Tannenbaum. On affine plane curve evolution. *Journal of Functional Analysis*, 119(1):79–120, 1994.

[57] B. Schölkopf and A. Smola. *Learning with kernels.* MIT Press, 2002.

[58] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit, Third Edition.* Kitware Inc.

[59] J. Sethian. A Review of Recent Numerical Algorithms for Hypersurfaces Moving with Curvature-Dependent Speed. *Journal of Differential Geometry*, 31:131–161, 1989.

[60] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science.* Cambridge University Press, June 1999.

[61] C. E. Shannon. *A Mathematical Theory of Communication.* CSLI Publications, 1948.

[62] M. Spivak. *Comprehensive Introduction to Differential Geometry (Volume 2).* Publish or Perish, 2nd edition.

[63] R. Summers. CT colonography computer-aided detection: Effect on radiologist observers. In *MICCAI*, 2010.

[64] R. Summers, J. Yao, P. Pickhardt, M. Franaszek, I. Bitter, D. Brickman, V. Krishna, and R. Choi. Computed tomographic virtual colonoscopy computer-aided polyp detection in a screening population. *Gastroenterology*, 129(6):1832–44, 2005.

[65] P. Sundaram, A. Zomorodian, C. Beaulieu, and S. Napel. Colon polyp detection using smoothed shape operators: preliminary results. *Med Image Anal*, 12(2):99–119, 2008.

[66] K. Suzuki, H. Yoshida, J. Näppi, S. G. Armato, and A. H. Dachman. Mixture of expert 3D massive-training ANNs for reduction of multiple types of false positives in CAD for detection of polyps in CT colonography. *Medical Physics*, 35(2):694, 2008.

[67] S. Taylor, S. Halligan, D. Burling, M. Roddie, L. Honeyfield, J. McQuillan, H. Amin, and J. Dehmeshki. Computer-assisted reader software versus expert reviewers for Polyp Detection on CT Colonography. *Am. J. Roentgenol.*, 186(3):696–702, 2006.

[68] J.-P. Thirion and A. Gourdon. Computing the differential characteristics of isointensity surfaces. *Computer Vision and Image Understanding*, 61(2):190 – 202, 1995.

[69] C. van Wijk, V. F. van Ravesteijn, F. M. Vos, and L. J. van Vliet. Detection and segmentation of colonic polyps on implicit isosurfaces by second principal curvature flow. *IEEE Trans Med Imaging*, 29(3):688–698, 2010.

[70] D. Vining, Y. Ge, D. Ahn, and D. Stelts. Virtual colonoscopy with computer-assisted polyps detection. *Computer-Aided Diagnosis in Medical Imaging Elsevier Science B. V.*, pages 445–452, 1999.

[71] D. Vining, D. Gelfand, R. Bechtold, E. Scharling, E. Grishaw, and R. Shifrin. Technical feasibility of colon imaging with helical CT and virtual reality. *Am J Roentgenol*, 162(Suppl), 1994.

[72] M. P. Wand and M. C. Jones. *Kernel Smoothing (Monographs on Statistics & Applied Probability).* Chapman and Hall/CRC, Dec. 1994.

[73] Z. Wang, Z. Liang, L. Li, X. Li, B. Li, J. Anderson, and D. Harrington. Reduction of false positives by internal features for polyp detection in CT-based virtual colonoscopy. *Med Phys*, 32(12):3602–16, 2005.

[74] R. A. Wolber and D. A. Owen. Flat adenomas of the colon. *Hum Pathol*, 22(1):70–4, 1991.

[75] World Health Organization. Cancer, http://www.who.int/mediacentre/factsheets/fs297, 2011.

[76] H. Yoshida and J. Näppi. Three-dimensional computer-aided diagnosis scheme for detection of colonic polyps. *IEEE Trans Med Imaging*, 20(12):1261–74, 2001.