Illumination Correction and Analysis of Two-Dimensional Microscopy Images of *Loa loa* Microfilariae

Faroq Mohammed Musaed AL-Tam

Tese de Doutoramento em Engenharia Informática

Trabalho efetuado sob a orientação de:
Hamid Reza Shahbazkia e António dos Anjos

2016

Statement of Originality

Illumination Correction and Analysis of Two-Dimensional Microscopy Images of *Loa loa* Microfilariae

Statement of authorship: The work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text. The material has not been submitted, either in whole or in part, for a degree at this or any other university.

Candidate: ______________________________

(Faroq AL-Tam)

"*Life can only be understood backwards; but it must be lived forwards.*"

Søren Kierkegaard

UNIVERSITY OF ALGARVE

Abstract

Faculty of Sciences and Technology
Department of Electronic and Informatics Engineering

by Faroq AL-Tam

This thesis addresses the problem of detecting a common parasitic microfilaria that causes loaisis, a major disease problem in Central and Western Africa. The dose of medicine to be administered to the patient is proportional to the estimated number of microfilariae in the patient's body. Therefore, proper estimation of the number of microfilariae is the key for conducting the right procedure. The clinical examination is necessary to estimate the microfilariae density in a blood sample drawn from the patient. Thereafter, visual inspection of the sample is performed.

The main challenge in this work is, however, the development of an automatic detection system of microfilariae in 2-D images. Such problem is new in the image processing literature, and the development of such system is very important for performing better diagnosis and treatment of this disease and other similar diseases.

A comprehensive review of, both generic and thin, object detectors in 2-D images is presented. A very robust method for microscopy image illumination correction is proposed, and a new powerful descriptor, the Hessian-Polar Context (HPC), for microfilariae is also introduced. These are then combined in a microfilariae detection system, where a simple, yet efficient, hypotheses generator is also presented. Additionally, several methods and applications for different image modalities are proposed. These involve a method and an application for the analysis of rice panicle in 2-D images. Additionally, an efficient method for artifact suppression in X-ray image is also proposed.

The proposed methods are compared to a set of state-of-the-art methods. Experimental results show that the developed methods are great contributions to the microscopy and X-ray imaging fields.

UNIVERSIDADE DO ALGARVE

Resumo

Faculdade de Ciências e Tecnologia
Departamento de Engenharia Eletrónica e Informática

por Faroq AL-Tam

Esta tese aborda o problema da deteção de um parasita comum, microfilaria, responsável pela doença Loaíase com elevada incidência na África Central e Ocidental. A dose da terapêutica a administrar a um doente depende do grau de infeção e é proporcional ao número estimado de microfilariae presente no seu organismo, sendo este número a chave para o procedimento adequado.

A densidade de microfilariae numa amostra de sangue é analisada visualmente, isto é, uma inspeção que consiste numa contagem manual. O maior desafio deste trabalho prende-se com o desenvolvimento de um sistema de deteção automática de microfilariae em imagens 2-D.

Este problema é novo na literatura que aborda o processamento de imagens. O seu desenvolvimento é fundamental para se conseguir obter um diagonóstico mais rigoroso e consequentemente decidir qual o melhor tratamento a aplicar para a doença Loaíses e outras doenças da mesma natureza. Com o intuito de atingir os objetivos pretendidos, foi feita uma revisão literária exaustiva sobre o problema da deteção de objetos em imagens a duas dimensões. Esta revisão incluiu tanto os métodos para objetos em geral bem como os métodos mais específicos para objetos estreitos. Como resultado do trabalho desenvolvido nesta tese, apresenta-se um método novo, muito robusto, para a correção da iluminação de imagens microscópicas através do ajuste iterativo de superfícies. Apresenta-se ainda, um novo descritor para microfilariae muito potente, denominado Hessian-Polar Context (HPC), utilizado em conjunto com algoritmos de aprendizagem máquina. Propõe-se também neste trabalho um gerador de hipóteses, extremamente eficiente apesar da sua simplicidade. A combinação do HPC com o gerador de hipóteses forma um sistema de deteção de microfilariae muito eficaz e de grande utilidade. Este sistema é dotado de uma interface gráfica muito intuitiva e implementada de forma modular.

Durante o desenvolvimento deste trabalho, investigaram-se temas paralelos e problemas similares, o que levou ainda à criação de vários métodos e aplicações inovadoras. Dada a sua relevância e proximidade com os problemas em análise, foram também incluídos nesta tese.

Destacam-se, de entre estes, a criação de um método e aplicação para a fenotipagem de plantas do arroz, e a criação de um método extremamente eficaz na correção de artefactos em imagens resultantes da tomografia computadorizada de carnes. Os métodos propostos foram testados e comparados com outros métodos estadodaarte, e os resultados demonstram que os propostos são efetivamente contribuições de enorme importância no campo da visão computacional e, mais especificamente, no que diz respeito às imagens microscópicas e de raio-X.

To my family

وسام و إيليا

Acknowledgements

I will start by thanking my advisers António dos Anjos, for his continuous support in every step I made towards the accomplishment of this work, and Hamid Reza Shahbazkia, for being the major responsible for my engagement in this research work and his continuous encouragement, support, and suggestions.

Without the support of my wife, my mother and father, it would not be easy to finish this work. I love you guys.

لكم كل الحب والإمتنان

A very special thanks goes to Ofélia Ponte and Jorge Andrez Malveiro. They know why.

I would also like to thank Thomas Martini Jørgensen, Bjarne Ersbøll, Hans du Buf, and Vedrana Andersen Dahl for the interesting discussions we had and all their support.

Many thanks to Denise Candeias, for all the proof-reading and useful comments.

I would like to acknowledge my home university, Thamar University-Yemen, and IRD-France that sponsored the work in this thesis, through my doctoral grant from Thamar University (No. 80 of the year 2012), and the partial support from Infectiopôle Sud, Marseille-France, and the Bill & Melinda Gates foundation.

A final thanks goes to Pedro Guerreiro, director of the PhD program, and the University of Algarve for allowing me to complete another stage of my academic life.

CONTENTS

List of Tables xix

List of Figures xxi

1 Introduction 1
- 1 Microfilaria Image Analysis . . . 1
- 2 *L. loa* Microfilariae . . . 3
- 3 Problems Associated with *L. loa* Images . . . 3
- 4 Thesis Focus and Contribution . . . 6

2 Principles of Image Analysis 7
- 1 Introduction . . . 7
- 2 Digital Image . . . 8
- 3 Image Histogram . . . 12
  - 3.1 Histogram Measures . . . 14
    - 3.1.1 χ^2 Distance . . . 15
    - 3.1.2 Minkowski Distance(norms) . . . 15
- 4 Image Enhancement . . . 15
  - 4.1 Spatial Domain Filtering . . . 16
  - 4.2 Optimization Based . . . 18
  - 4.3 Frequency Domain Filtering . . . 19
    - 4.3.1 Wavelets . . . 19
    - 4.3.2 Curvelets . . . 20
- 5 Image Segmentation . . . 22
  - 5.1 Thresholding . . . 23
    - 5.1.1 Global Thresholding . . . 24
    - 5.1.2 Local Thresholding . . . 24

5.2 Edge Based Segmentation 26
5.2.1 Image Derivatives 27
5.2.1.1 Laplacian of Gaussian 30
5.2.1.2 Gaussian Derivatives 31
5.3 Clustering 34
5.4 Graph Cut 35
5.5 Region Growing 36
5.6 Active Contours 37
6 The Hessian 40
7 Connected Component Analysis 43
7.1 Connected Components 43
7.2 Distance Map 43
7.3 Medial Axis Transform 45

3 Illumination Correction **47**
1 Introduction 47
2 Related Work 48
3 The Proposed Approach 51
3.1 Sampling with Dynamic Programming (DP) 51
3.2 Sampling Using the Watershed Transform 55
3.3 Algorithm 56
3.3.1 Estimating the Illumination Field 56
3.3.2 Correcting the Bias 57
3.3.3 Regularization 58
3.3.4 Stopping Criterion 59
4 Experimental Results 61
4.1 Materials 61
4.2 Evaluation 62
4.3 Discussion 64
5 Conclusion 66

4 Object Detection **69**
1 Introduction 69
2 Feature Extraction 73
2.1 Scale-space 75
2.2 Keypoints and Blobs Detectors 75
2.2.1 Harris and Hessian Detectors 76
2.2.2 Maximally Stable Extremal Regions 79
2.3 Global Features 80
2.3.1 Histogram Pooling 80
2.3.2 Co-occurrence Matrix 81
2.3.3 Histogram of Oriented Gradients 82
2.4 Local Features 83
2.4.1 Scale Invariant Feature Transform 84

2.4.1.1 Detecting Keypoints 84
2.4.1.2 Orientation of Keypoints 86
2.4.1.3 Calculating the Descriptors 86
2.4.2 Rotation Invariant Feature Transform 87
2.4.3 Shape Context 88
2.4.4 Basic Image Features 88
2.4.5 Local Binary Pattern 89
2.4.6 Binary Robust Independent Elementary Features . . 90
2.4.7 Local Intensity Order Pattern 91
2.4.8 Others 92
3 Model Building 93
3.1 Support Vector Machines 93
3.1.1 Handling Non-linear Data 97
3.2 Soft Margin C-SVM 97
4 Hypothesis Generation 98
4.1 Sliding Window 99
4.2 Voting 100
4.3 Segmentation 102
4.4 Other Approaches 103
5 Hypothesis Verification 104
6 Conclusions 105
6.1 Detection of Features 105
6.2 Model Building 106
6.3 Hypothesis Generation 107
6.4 Hypothesis Verification 108

5 Thin Structure Detection 111
1 Introduction 111
2 Matched Filters 112
3 Differential Geometry 113
3.1 Frangi's Multi-scale Filtering 114
3.2 Steger's Method 114
4 Local Elevations Detection 115
5 Oriented Morphology 118
5.1 Path Opening 118
5.2 Morpho-Hessian 119
6 Path Finding 119
7 Tracking 120
8 Binary Descriptor 121
9 Conclusion 123

6 Microfilaria Detection with the Hessian-Polar Context 127
1 Introduction 127
2 Hypothesis Generation 128

3 Hessian-Polar Context Descriptor . . . 131
3.1 Medial Points' Detection . . . 131
3.2 Local Radius Estimation . . . 132
3.3 Orientation Estimation . . . 133
3.4 Implementation of HPC . . . 133
3.4.1 Dimensionality Reduction and Normalization . . . 137
3.4.2 Incorporating Color . . . 138
3.4.3 Multi-classifier Ensembling . . . 139
3.4.4 Complexity . . . 139
4 Experimental Results . . . 140
4.1 Materials . . . 140
4.2 Annotation . . . 142
4.3 Parameters . . . 143
4.4 Comparing the Descriptors . . . 144
4.4.1 Generic Features Only . . . 144
4.4.2 Incorporating Color . . . 147
4.4.3 Set Testing . . . 149
4.5 Comparison with "the" State-of-the-art Detector . . . 154
5 Conclusions . . . 158

7 Applications **161**
1 Introduction . . . 161
2 Panicle and Grain Analysis . . . 162
2.1 Background . . . 162
2.2 P-TRAP: a Panicle Trait Phenotyping Tool . . . 162
3 X-ray Rings Artifact Suppression . . . 163
3.1 Rings Artifacts . . . 165
3.2 Scale-dependent Suppression . . . 166
3.3 Evaluation . . . 168
4 *In situ* Microfilariae Analysis . . . 170

8 Conclusion **173**

A Integral Image **177**

B Features Pooling Using Bag-of-Words **179**

Bibliography **181**

LIST OF TABLES

3.1 Mean and Standard Deviation of the illumination correction results. . 63

6.1 Measures used in the evaluation. 141
6.2 Averaged 10-fold cross-validations binary classification measures. . . . 145
6.3 Averaged measures for 10-fold cross validations for combined features. 148
6.4 Testing results for training with sets A and B and testing set C. . . . 150
6.5 Testing results for training with sets A and C and testing set B. . . . 150
6.6 Testing results for training with sets B and C and testing set A. . . . 151
6.7 Detection results. HPC on the left and part-based HOG on the right. 156

LIST OF FIGURES

1.1 A typical microfilaria. 4
1.2 A giemsa-stained *L. loa* microfilaria. 4
1.3 Sample microfilariae images with illumination bias. 5
1.4 Overlapped microfilariae. 5
1.5 Microfilaria-like debris. 5
1.6 Microfilariae contaminated by debris. 5

2.1 A digital grayscale image represented by a matrix. 9
2.2 A digital color image and its RGB channels. 11
2.3 A binary image. 11
2.4 The neighborhood system. 12
2.5 Image histogram. 12
2.6 Noise filtering in a grayscale image. 17
2.7 Image thresholding using different approaches. 25
2.8 Local thresholding. 27
2.9 A typical step edge and its first and second order derivatives. 28
2.10 Directional derivatives of an image. 28
2.11 Gradient magnitude and orientation of an image. 29
2.12 The LoG filter. 32
2.13 Edge detection. 34
2.14 Image segmentation by clustering. 35
2.15 Image segmentation using grow-cut. 37
2.16 The surface plot of an Image. 39
2.17 A 2-D Gaussian function. 40
2.18 Level set active contour segmentation. 41
2.19 Connected components of a binary image. 44
2.20 Euclidean Distance Map of a binary image. 44
2.21 Medial axis detected by various approaches superimposed on a sample. 46

3.1 Applying the dynamic programming approach to detect different features on a sample image. 54
3.2 Effect of smoothing on the results of the Watershed Transform. . . . 55
3.3 Iterative fitting by ever-increasing polynomial for 1D signal. 60
3.4 Different biases created by 2-D Gaussian functions and by 2-D polynomials. 62
3.5 RMSE between the tested methods and the ground truth. 63
3.6 Examples of the results of different methods. 65
3.7 The results of applying the proposed approach on microfilariae images. 66

4.1 Different objects. 70
4.2 Object detection. 71
4.3 Covariant regions detected by various detectors of the sample image. 78
4.4 The MSER of the sample image. 80
4.5 The HOG response of a sample image. 83
4.6 The SIFT keypoints and descriptors. 84
4.7 SIFT octaves. 85
4.8 Local Binary Pattern calculated for a sample. 90
4.9 Rotation invariant approach of LIOP. 92
4.10 SVM classification of two linearly separable classes with two features. 94
4.11 Candidate objects proposed by different approaches. 108

5.1 The bell-shaped profile of a cross-section of a microfilaria. 112
5.2 The response of the Matched Filter. 113
5.3 Ridges detected by Steger's approach. 115
5.4 The responses of several mathematical morphology operators applied to a sample image. 117
5.5 The surface plot of the top-hat transform applied to a sample. 118
5.6 Histogram representation of synthetic objects. 122
5.7 The response of different detectors on overlapped microfilariae. 124
5.8 A microfilaria sample with a large artifact. 125

6.1 Object candidates produced by Selective Search and the proposed approach. 130
6.2 Calculating the Hessian-Polar Context descriptor. 132
6.3 The orientation field of the eigenvector associated with the smallest eigenvalue superimposed on a sample. 134
6.4 The HPC calculated for some points of two samples. 135
6.5 HPCs calculated for different microfilaria and non-microfilaria objects. 137
6.6 Set A. 141
6.7 Set B. 142
6.8 Set C. 142
6.9 An annotated sample. 143
6.10 ROC Curve for the generic descriptors. 146
6.11 PR Curve for the generic descriptors. 147

6.12 ROC Curve for the combining shape and color features. 148
6.13 PR Curve for the shape and color descriptors. 149
6.14 ROC Curve for training using sets A and B and testing set C. 151
6.15 PR Curve for training using set A and B and testing set C. 152
6.16 ROC Curve for training using set A and C and testing set B. 152
6.17 PR Curve for training using set A and C and testing set B. 153
6.18 ROC Curve for training using set B and C and testing set A. 153
6.19 PR Curve for training using set B and C and testing set A. 154
6.20 Microfilariae detection using part-based HOG. 155
6.21 Detection results. 157
6.22 False positive and false negative objects processed by HPC. 158
6.23 False positive and false negative objects processed by part-based HOG. 158

7.1 Panicle traits application. 163
7.2 A phase-contrast slice calculated from its sinogram. 164
7.3 Rings artifacts. 165
7.4 A phase-contrast slice transformed from the Cartesian to the polar domain. 166
7.5 Discrete curvelet transform of a sample image. 167
7.6 Rings' suppression results. 169
7.7 Rings' suppression results in different scales. 170
7.8 The GUI of the *In situ* microfilariae detector. 171
7.9 The selection of reference microfilariae. 172

A.1 Integral image calculation. 178

NOMENCLATURE

$\lceil \square \rceil$ Ceiling of $\square$.

Δ The Laplacian.

$\lfloor \square \rfloor$ Floor of $\square$.

$\frac{\partial \boldsymbol{A}}{\partial \square}$ Matrix partial derivative with respect to direction $\square$.

$\mathcal{A}$ A special matrix or function.

$\boldsymbol{A} = \{a_{i,j}\}$ matrix of elements $a_{i,j}$.

$\boldsymbol{A}^\top$ Matrix transpose.

$\boldsymbol{A}^{(k)}_{\alpha \times \beta}$ The kth sub-matrix of $\boldsymbol{A}$ such that $\boldsymbol{A}$ is subdivided with respect to $\alpha \times \beta$, where α and β are the number of rows and columns of the sub-matrix, respectively.

$\boldsymbol{A}_{m \times n}$ Matrix with m rows and n columns.

∇ The gradient.

$\|\boldsymbol{a}\|_1$ Manhattan norm ℓ_1, $\sum_i |a_i|$.

$\|\boldsymbol{a}\|_2$ Euclidean norm ℓ_2, $\sqrt{\boldsymbol{a}^\top \boldsymbol{a}}$.

$\mathrm{Tr}(\boldsymbol{A})$ The trace of $\boldsymbol{A}$, *i.e.* the sum of its main diagonal elements.

$\boldsymbol{a} = \left[a_1, \cdots\right]^\top$ A column vector.

$\boldsymbol{a}_i^\top$ The ith row of the matrix.

$\boldsymbol{a}_j$ The jth column of the matrix.

$diag(\square)$ A matrix with $\square$ as its main diagonal.

1

INTRODUCTION

1 Microfilaria Image Analysis

DIGITAL microscopy image analysis is a research field that arouses an increasing interest in the biomedical world. The amount of data obtained from biological experiments justifies the need for automatic processing. Common uses for this kind of analysis is the diagnosis of diseases and/or drug design. In this context, the manual analysis of microscopic particles is very time-consuming and error-prone (if not impossible). This work addresses the automatic analysis of a type of microscopic images used in the study and diagnosis of a common parasitic disease, loaisis, which is a major health problem in Central Africa.

Loiasis is caused by a filarial nematode (round worm) called *Loa loa* [Org97]. This disease affects over 12 million people in more than 11 African countries. The main vectors of *L. loa* are two species of tabanids: *Chrysops silacea* and *C. dimidiata*. The most frequent manifestations of loiasis are transient lymphoedemas (known as "Calabar swellings") usually occurring on the upper limbs, and the migration of the adult stage of the parasite under the conjunctiva of the eye (loiasis is also known as the "African eye-worm"). Adult worms live under the skin or between muscles, and females produce millions of embryos, called microfilariae, that invade the blood circulation from where they can be picked up by the vector to maintain the parasite cycle. In addition to being a health issue itself, loiasis is also a subject of interest because individuals harbouring high microfilarial densities, can develop serious adverse events (SAEs) after treatment with the drug ivermectin [GGWDN+97, BGGWC03, Bou06]. Ivermectin is widely used to control two other filarial diseases which constitute public health problems in Africa: onchocerciasis known as "river blindness" and lymphatic filariasis known as "elephantiasis".

Although the clinical manifestations mentioned above, are pathognomonic, the standard method to diagnose *L. loa* infection is by microscopic examination of the blood smears prepared with blood drawn from the host. This kind of diagnosis involves a process of staining and optional concentration to demonstrate the presence of microfilariae in the blood. In most cases, giemsa-staining is used, and when possible, the Knott's technique using formalin can be applied for concentration [Kno39]. However, the microscopy slides may contain debris and unwanted particles (dust, etc.). Moreover, images are often affected by a biased illumination field when improper microscope settings are used.

2 *L. loa* Microfilariae

L. loa filarial worms, like other lymphatic worms develop and grow inside the human body. *L. loa* adult worms inhabit subcutaneous tissues and conjunctivae, which make them difficult to identify visually. Therefore, it is important to identify them when they are microfilariae, *i.e.* at the larval phase.

A microfilaria is a primitive serpentine organism covered by sheath and contains many cells nuclei (Figure 1.1). Its length varies from 231-300 μm and the width ranges from 5 to 7 μm. It is worth noting that, *L. loa* microfilariae's morphological features distinguish these from other similar microfilariae. *L. loa* microfilariae show a single row of nuclei to the end of the tail, and near the tail the nuclei are spaced irregularly [Org97]. In addition, they have a tapered tail, and their sheath does not appear in the giemsa-staining. In fact, when imaged, *L. loa* microfilariae appear as nematode-like objects (Figure 1.2).

The change in the number of circulating microfilariae in the host's peripheral blood is called the periodicity. *L. loa* microfilariae are diurnally periodic, which means they only exist in the peripheral blood at certain daytime hours, compared to nocturnally periodic, *e.g.* Wuchereria bancrafti, which can only be found during the nighttime. Because of periodicity change from one creature to another, each patient is examined in 2 to 4 hour intervals during a period of 24 to 30 hours.

3 Problems Associated with *L. loa* Images

Detecting microfilariae in 2-D images is not an easy task. Herein, some common problems are presented, and these will be tackled separately.

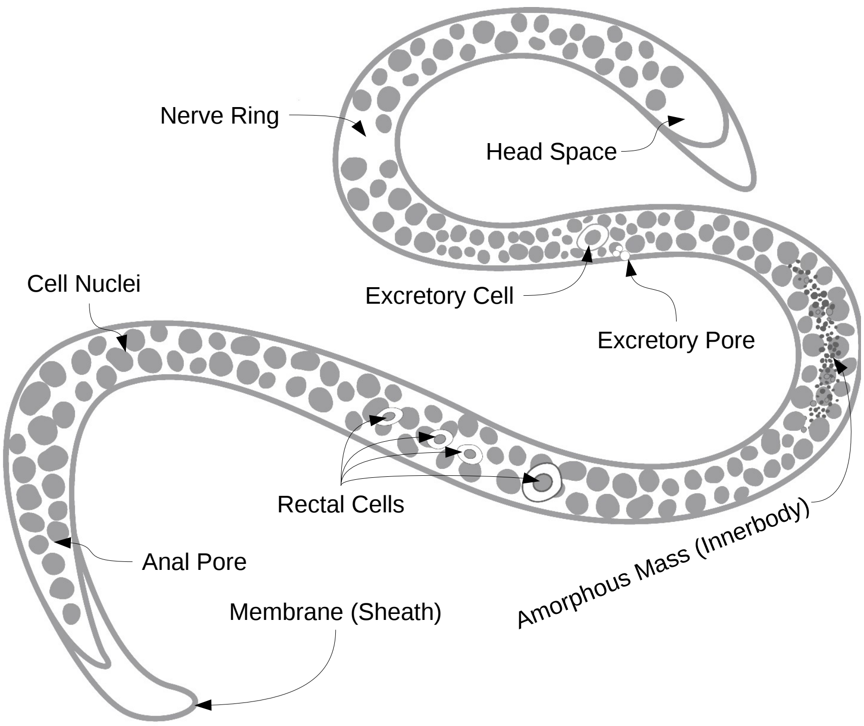


Figure 1.1: A typical microfilaria.

Figure 1.2: A giemsa-stained *L. loa* microfilaria.

One of the main problems is the illumination bias, where images are contaminated by uneven illumination. This problem is covered in depth in Chapter 3, but some examples are shown in Figure 1.3.

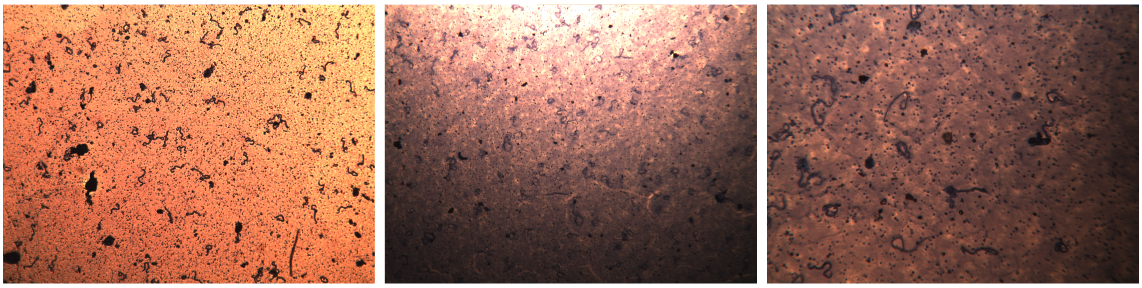


Figure 1.3: Sample microfilariae images with illumination bias.

Microfilariae are elastic objects. They can fold and overlap in almost a free-form shape as depicted in Figure 1.4. In addition, images are contaminated by debris, which, in some cases, form microfilaria-like objects (Figure 1.5). In some other cases, these artifacts hide parts of the microfilariae as shown in Figure 1.6. These problems will be discussed in details in the next few chapters.

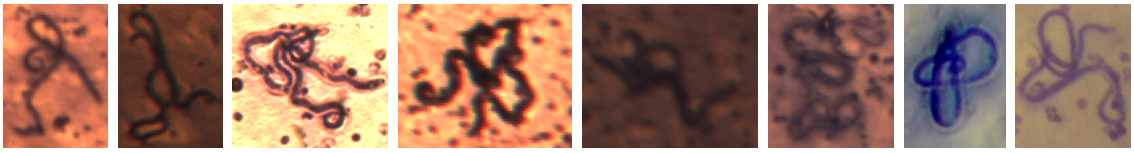


Figure 1.4: Overlapped microfilariae.

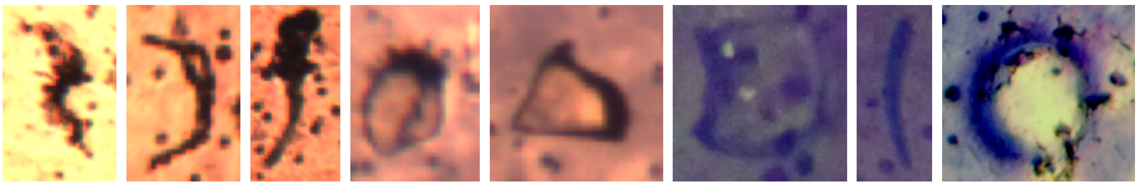


Figure 1.5: Microfilaria-like debris.

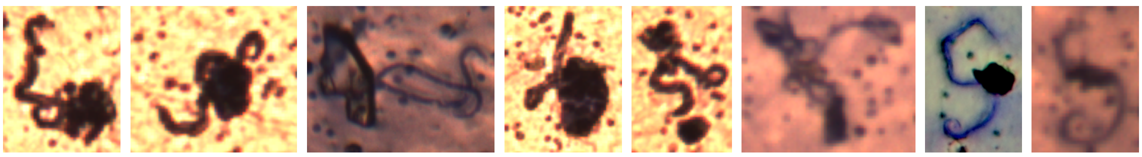


Figure 1.6: Microfilariae contaminated by debris.

4 Thesis Focus and Contribution

This thesis is organized as follows. Chapter 2 includes an introduction to image processing principles used in this work. The proposed illumination correction approach is presented in Chapter 3. Chapters 4 and 5 discuss the problems of generic object and thin structure detection, respectively. The proposed detection approach as well as the experimental results are presented in Chapter 6. In Chapter 7, some methods developed during the work in this thesis are presented. Finally, the thesis is concluded in Chapter 8.

The objective of this work is to develop an approach to identify microfilariae in giemsa-stained images. Such a technique would be most useful to do rapid in routine identification of the individuals who have high microfilarial densities, and thus susceptible to developing a post-ivermectin SAE. Using a test-and-treat strategy, individuals at risk (who represent less than 5% of the total infected population) could be excluded from mass treatment with ivermectin, whereas the rest of the population could be safely treated with the drug. In essence, the main contributions in this thesis are:

- A new illumination correction approach for microscopy images.
- A comparison of well-established image descriptors for microfilariae detection.
- A new descriptor for thin objects.
- A detection system for microfilariae in 2-D images.
- A new method for rings artifacts suppression in CT volumes.
- Several applications in microscopy and plant analysis.

2

PRINCIPLES OF IMAGE ANALYSIS

"The image is more than an idea. It is a vortex or cluster of fused ideas and is endowed with energy."

– Ezra Pound, American poet and critic.

1 Introduction

A microscopy specimen can be represented in four main modalities: optical, continuous, digital, and displayed [WMC10]. Each one of them is assumed to be a perfect representation of the specimen which, in fact, is not the case due to noise and other imaging and digitization artifacts. The microscope optics project the specimen onto a charge-coupled device (CCD). When the photo-active layer (epitaxial layer of silicon) of the CCD receives the projected light from the lens, each capacitor accumulates an electrical charge proportional to the received light intensity. Each capacitor transfers its charge to its neighbor. The charge received at the last capacitor is amplified and converted into voltage. Accordingly, the input image is converted into a sequences of voltages. Mathematically, the optical image can be represented as a 2-D continuous function, known as the continuous image. However, for this image to be stored in the computer memory, it has to be converted into a digital image via sampling and quantization. The displayed image is the specimen representation in the computer screen.

2 Digital Image

A digital image is represented as a grid of pixels as shown in Figure 2.1, where the origin of the xy-coordinate system is located at the top left cell. Each cell represents a pixel (picture element). The grayscale and color images are two of the most common types of digital images.

The image shown in Figure 2.1 is grayscale and the pixels represent the luminance. A grayscale image is mathematically defined as a function $\boldsymbol{S} : \Omega \subset \mathbb{R}^2 \longrightarrow [0, l-1]$, where Ω is the image domain (usually a rectangle), and $l = 2^k$ is the number of grayscale levels. For an 8-bit grayscale image (*i.e.* k=8) each pixel takes one byte, meaning that, there are $l = 256$ different intensity levels. In the present thesis, the observed grayscale image $\boldsymbol{S}$ will be treated as a matrix of size $m \times n$, and is modeled as [You01, GW02]:

$$\boldsymbol{S} = \boldsymbol{T} \odot \boldsymbol{B} + \boldsymbol{\epsilon}, \tag{2.1}$$

where $\boldsymbol{T}_{m\times n}$ is the true image, $\boldsymbol{B}_{m\times n}$ can be any multiplicative noise matrix (*e.g.* an illumination field), $\odot$ is the Hadamard product, and $\boldsymbol{\epsilon}$ is the additive noise.

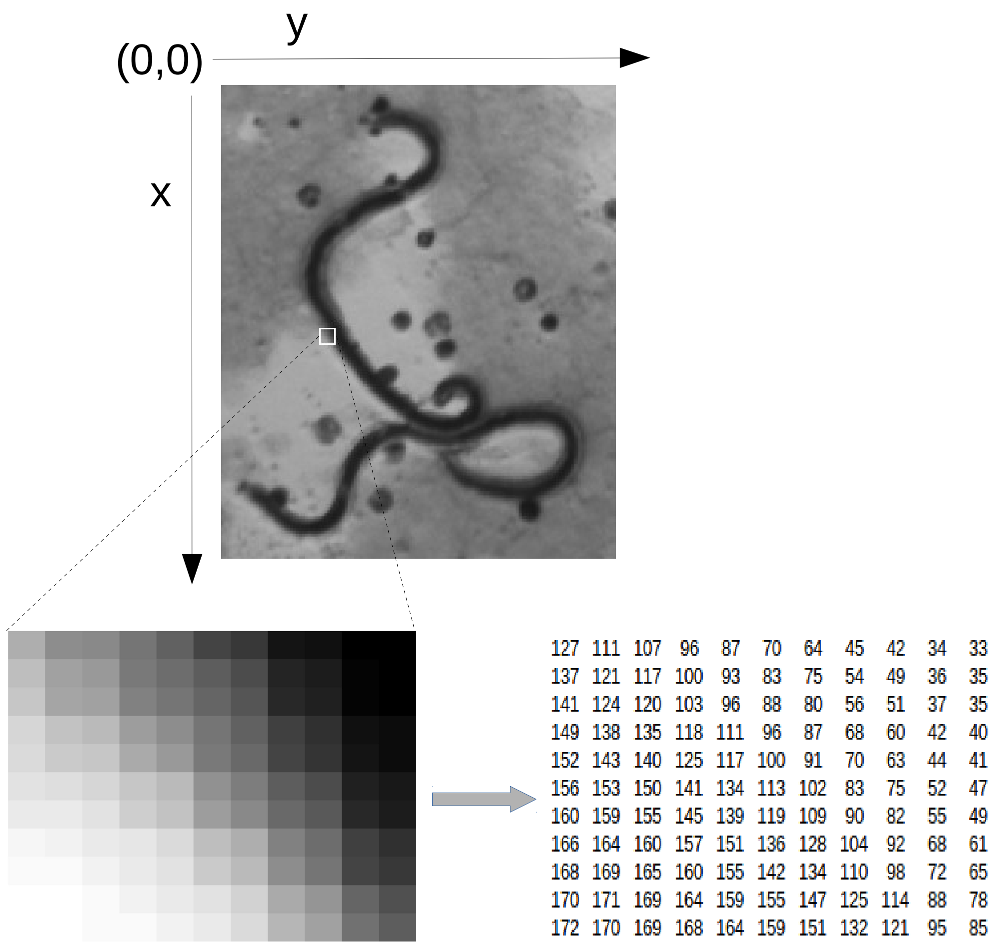


Figure 2.1: A digital grayscale image represented by a matrix.

Once the image is represented as a matrix (Figure 2.1), the matrix coordinate system is used, which consists in the 90° clockwise rotation of the 2-D Cartesian coordinate representation [Jai89].

In color images, each pixel represents information gathered from different channels, *e.g.* Red, Green, and Blue (RGB). Therefore, color images may be represented using the compact representation, where a pixel encodes the RGB values of a position in a single value. Individual grids can also be used, where each grid represents a color channel, such that, the image is a 3-D matrix $\boldsymbol{S}^{(\mathrm{RGB})}$ of size $m \times n \times 3$, which can be decomposed into three matrices of size $m \times n$: $\boldsymbol{S}^{(\mathrm{red})}$, $\boldsymbol{S}^{(\mathrm{green})}$, and $\boldsymbol{S}^{(\mathrm{blue})}$ for the red, green, and blue channels, respectively. The RGB image is then defined as the mapping: $\boldsymbol{S}^{(\mathrm{RGB})} : \Omega \subset \mathbb{R}^2 \longrightarrow \Gamma \subset \mathbb{R}^3$, where Γ is the range that defines the 3-D RGB color space. This representation for color images will be used in this thesis (Figure 2.2). The gray level image $\boldsymbol{S}$ of $\boldsymbol{S}^{(\mathrm{RGB})}$ can be calculated as:

$$\boldsymbol{S} = \alpha \boldsymbol{S}^{(\mathrm{red})} + \beta \boldsymbol{S}^{(\mathrm{green})} + \gamma \boldsymbol{S}^{(\mathrm{blue})} \tag{2.2}$$

where the parameters, $\alpha, \beta, \gamma > 0$ and $\alpha + \beta + \gamma = 1$.

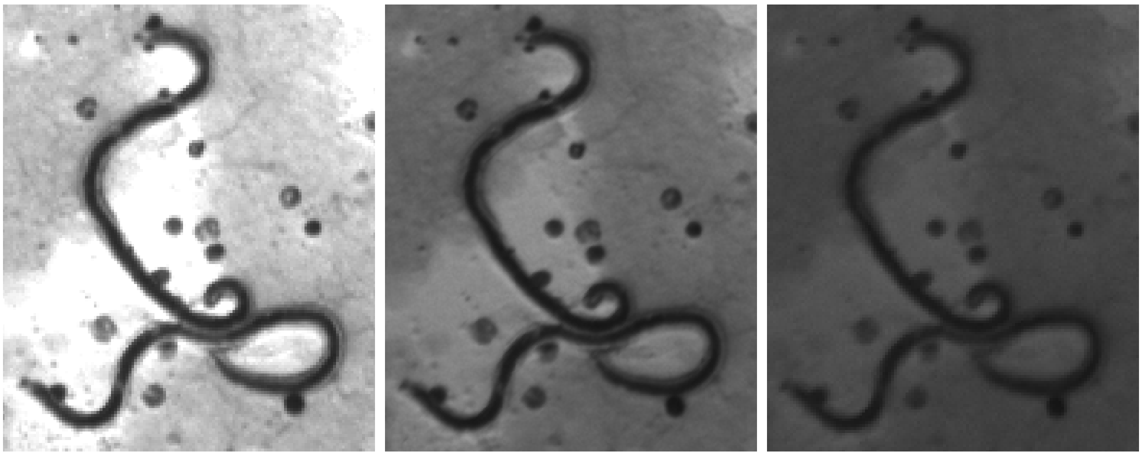
Besides grayscale and color, images can also be binary (Figure 2.3). A binary image can be defined as $\boldsymbol{S}^{(\mathrm{binary})} : \Omega \subset \mathbb{R}^2 \longrightarrow \{0, 1\}$, where 0 and 1 represent the background and the foreground, respectively. For an easier visualization, in this work, black represents the foreground and white represents the background in the figures. Binary images are simple and can therefore be represented in many different ways, such as matrices (used in this work), or as run-length encoding, which is a list of connected horizontal 1's. In this encoding, the start position and length or end position must be recorded. This yields a compact representation for binary images.

Since images are represented by grids, an important basic principle used in many image processing operations is the *neighborhood* system. The neighborhood system defines how pixels are associated in the grid. Let $\boldsymbol{p}$ and $\boldsymbol{q}$ be two pixels belonging to the same grid in the $\mathbb{Z}^d$, for any dimension d, there are two main systems to define their neighborhood $N(\boldsymbol{p})$:

- Von Neumann neighborhood:

$$N(\boldsymbol{p}) = \left\{ \boldsymbol{q} : \sum_{i=1}^{d} |p_i - q_i| = 1 \right\} \tag{2.3}$$

(a) A color sample.



(b) Red channel $\boldsymbol{S}^{(\text{red})}$. (c) Green channel $\boldsymbol{S}^{(\text{green})}$. (d) Blue channel $\boldsymbol{S}^{(\text{blue})}$.

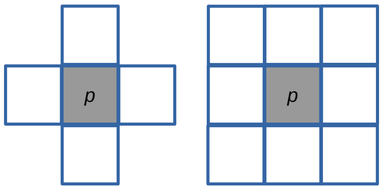
Figure 2.2: A digital color image and its RGB channels.

Figure 2.3: A binary version of the sample shown in Figure 2.1.

- Moore neighborhood:

$$N(\boldsymbol{p}) = \left\{ \boldsymbol{q} : \max_{i=1,\ldots,d} |p_i - q_i| = 1 \right\} \tag{2.4}$$

When $d = 2$, Equations 2.3 and 2.4 correspond to the *4-connected* and *8-connected* neighborhoods, respectively (Figure 2.4). Besides these two, hexagonal and octagonal neighborhood systems are also used in some applications.



(a) 4-connected. (b) 8-connected.

Figure 2.4: The neighborhood systems for a point $\boldsymbol{p}$.

3 Image Histogram

A histogram is a discrete function that pools observations into bins. Each observation class is represented by a bin in the histogram. The magnitude of each bin represents the number of occurrences of the observation. For a grayscale image, the bins represent the occurrences of each intensity in the image (Figure 2.5).

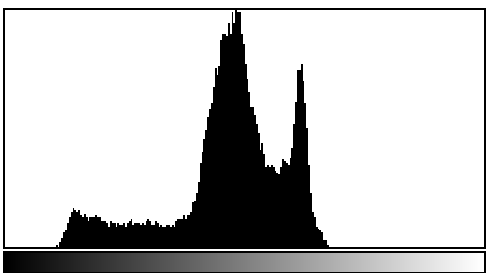


Figure 2.5: Histogram of the sample in Figure 2.1.

Formally, for an image $\boldsymbol{S}$ of size $m \times n$, the histogram is represented by:

$$\boldsymbol{h}(\boldsymbol{S}) = \sum_{x=1}^{m}\sum_{y=1}^{n}[s_{x,y} = i], \quad 0 \leq i \leq l-1 \tag{2.5}$$

where $[\square]$ is the Iverson bracket, *i.e.* $[\square] = 1$ if $\square$ is true and 0 otherwise. The pixel at position (x, y) is $s_{x,y}$.

Once the histogram is an approximation of the distribution of the intensities, other higher statistical measures, such as the mean and standard deviation, can be computed. The mean of $\boldsymbol{S}$ is:

$$\mu_{\boldsymbol{S}} = \frac{1}{mn}\sum ih_i(\boldsymbol{S}) \tag{2.6}$$

where h_i is the magnitude at bin i. The standard deviation of $\boldsymbol{S}$ is:

$$\sigma_{\boldsymbol{S}} = \sqrt{\frac{1}{mn-1}\sum\left(i-\mu_{\boldsymbol{S}}\right)^2 h_i(\boldsymbol{S})} \tag{2.7}$$

The probability that a given pixel has the intensity i is:

$$\Pr(h_i(\boldsymbol{S})) = \Pr(s_{x,y} = i) = \frac{h_i(\boldsymbol{S})}{mn} \tag{2.8}$$

and the entropy of $\boldsymbol{S}$ is then:

$$E_{\boldsymbol{S}} = -\sum_i \Pr(h_i(\boldsymbol{S}))\log\Pr(h_i(\boldsymbol{S})) \tag{2.9}$$

3.1 Histogram Measures

Histograms' comparison is a common problem in computer vision applications. It depends on three main factors: the nature of the histogram, alignment, and sparsity. When histograms are treated as vectors, Minkowski distances (norms) are used. Set theory can be used for comparison when bins are treated as elements and histograms are treated as sets. Bhattacharyya, Kolmogorov Smirnov, χ^2, and other distribution measures are used when considering histograms as discrete distributions. In information theory, histograms' comparison can be considered as an encoding problem, and can also be regarded as a discrete optimization problem, where the cost is the energy needed to morph a histogram into another, as in Earth Mover's Distance (EMD).

Bin-to-bin matching is sufficient when histograms have aligned domains and, in this case, the cost function is based on the aggregated error of the bin-wise comparison. If distributions are scaled due to lightness change, for example, histograms will have different domains and in this case a cross-bin measure is preferred.

Another consideration is whether the histogram is sparse or dense, because some measures are not suitable for sparse histograms' comparison. For more details on the types of histograms and methods of comparison, the reader is refereed to [CS02]. The following subsections discuss the measures used in this thesis.

3.1.1 χ^2 Distance

The χ^2 is perhaps the most common histogram measure. Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be two histograms, both of length k, the $\chi^2(\boldsymbol{u}, \boldsymbol{v})$ is defined as:

$$\chi^2(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2} \sum_{i=1}^{k} \frac{(u_i - v_i)^2}{u_i + v_i} \tag{2.10}$$

3.1.2 Minkowski Distance(norms)

Besides χ^2, Minkowski distance ($\ell_d(\boldsymbol{u}, \boldsymbol{v}),\ d = \{1,\ 2\ , \ldots,\ \infty\}$) for two histograms $\boldsymbol{u}$ and $\boldsymbol{v}$ is also popular for vector comparison. For a given degree d, this measure is defined as:

$$\ell_d(\boldsymbol{u}, \boldsymbol{v}) = \left(\sum_{i=1}^{k} |u_i - v_i|^d \right)^{\frac{1}{d}} \tag{2.11}$$

4 Image Enhancement

The quality of the obtained image relies on both the imaging devices and the personnel's expertise, which are not always perfectly fulfilled. Noise, imaging artifacts, and shading are common problems widely found in images. For further processing, images are required to undergo some preprocessing steps. This section addresses common noise filtering approaches. Fortunately, in most of the cases, basic spatial filtering methods, including median, Gaussian, and so on, are effective to reduce noise. However, in some cases, anisotropic methods are required to avoid smoothing objects' borders. In addition, some types of noise (*e.g.* pattern noise) can be better handled in the frequency domain, where wavelet and curvelet approaches have shown to be versatile tools [SMF10].

4.1 Spatial Domain Filtering

Perhaps, the easiest way to deal with noise, is to identify its type, and then use a convolution filter to remove it. The convolution operation denoted by $*$ can be simply viewed as the process of replacing the pixel under processing by the sum of the element-wise product between the flipped filter and the corresponding neighborhood in the image. Formally, when convolving $\boldsymbol{S}$ with a kernel $\boldsymbol{G}_{\alpha\times\beta}$, the new pixel at (x, y) position will be:

$$s_{x,y} = \sum_{i=-a}^{i=a} \sum_{j=-b}^{j=b} s_{x-i,y-j} \times g_{i+a,j+b} \tag{2.12}$$

where $a = \frac{\alpha-1}{2}$, $b = \frac{\beta-1}{2}$, and g is an element of $\boldsymbol{G}$.

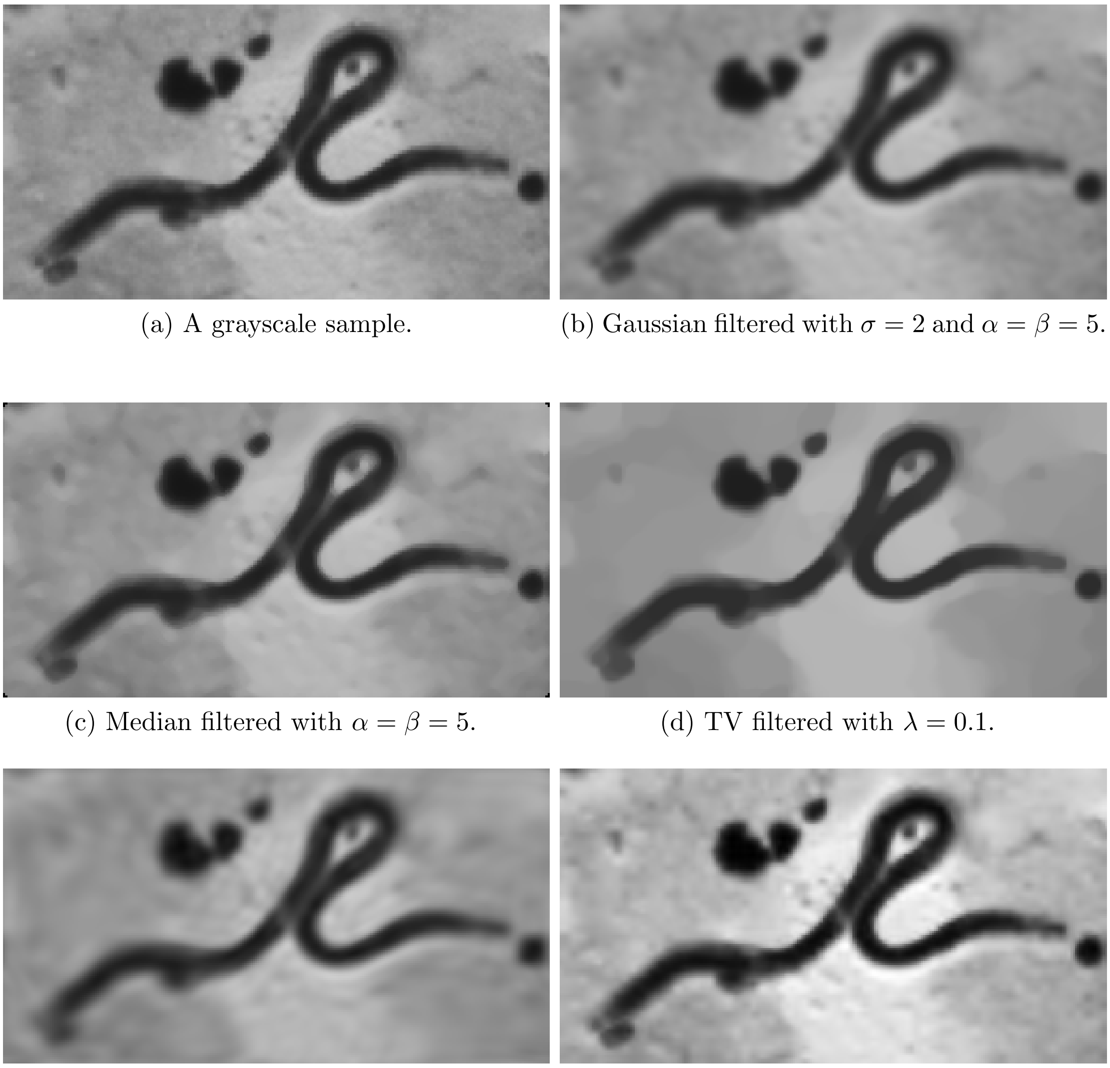
At the borders of the image, Equation 2.12 will be invalid. Therefore, the borders of $\boldsymbol{S}$ need to be handled. Two common approaches are padding a predefined value (*e.g.* zero) or repeating the border pixels.

A common convolution kernel is the Gaussian $\mathcal{G}$, which is a separable filter, and can be calculated as the discretization of the 2-D Gaussian function (Figure 2.17):

$$\mathcal{G}(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.13}$$

where σ is the standard deviation. In Section 6 and later in Chapter 4, it will be shown that, this kernel is helpful not only in filtering noise, but also in calculating image derivatives.

In addition, the median filter is also a common filter, which replaces the pixel under process by the median of its neighbors. An example of a grayscale sample filtered by Gaussian and median filters is shown in Figures 2.6b and 2.6c, respectively.



(a) A grayscale sample.

(b) Gaussian filtered with $\sigma = 2$ and $\alpha = \beta = 5$.

(c) Median filtered with $\alpha = \beta = 5$.

(d) TV filtered with $\lambda = 0.1$.

(e) Curvelets filtered.

(f) Wavelets filtered.

Figure 2.6: Noise filtering in a grayscale image.

4.2 Optimization Based

Although spatial low-pass filters are easy to implement, they do not handle object's borders properly. These approaches may affect the objects' boundaries and make it difficult to extract the foreground pixels accurately. For example, the Gaussian and mean low-passing filters are known to reduce the effect of high frequency noise but, at the same time, they affect the sharp components (edges) of the objects (see Figures 2.6b and 2.6c). To overcome this problem, other techniques such as Anisotropic Diffusion (AD) [PM90] or Total Variation (TV) denoising [ROF92] can be used. The objective is to find a smoother image while preserving the objects' boundaries intact. Assuming additive noise as in Equation 2.1, TV image denosing seeks a minimizer for the functional:

$$J^{(\text{TV-denoise})}(\boldsymbol{S};\lambda) = \int_{\Omega} |\nabla \boldsymbol{S}| + \frac{\lambda}{2}\int_{\Omega} (\boldsymbol{S} - \hat{\boldsymbol{S}})^2 \tag{2.14}$$

where λ is a positive parameter, Ω is the image domain, $\nabla \boldsymbol{S}$ is the gradient of $\boldsymbol{S}$ (see Section 5.2.1), and $\hat{\boldsymbol{S}}$ is the filtered version of $\boldsymbol{S}$. The first term in Equation 2.14 is referred to as the TV, which is a global measure of the oscillations (variations) in the image. Smooth images have a lower TV than sharp ones. The second term is the fidelity between the original and the filtered images. The parameter λ controls the degree of balance between denoising the image and preserving its details. The result of this method is shown in Figure 2.6d. It is clear that, this method is more effective in what processing edges of the object is concerned, when compared to the other approaches.

4.3 Frequency Domain Filtering

Since images are 2-D signals, frequency domain is helpful in detecting specific signals, such as noise, in order to remove them. Recently, wavelet and curvelet transforms are showing great importance in noise cancellation.

4.3.1 Wavelets

A discrete signal $f(\boldsymbol{t})$ can be approximated by a linear combination of orthogonal basis functions as [MTMS09a]:

$$f(\boldsymbol{t}) = \sum_{k=1}^{K} \alpha_k \psi_k(\boldsymbol{t}) \tag{2.15}$$

where $\boldsymbol{t}$ is the time domain variable and K is a scalar. α and ψ are the coefficients and the basis function, respectively.

In the Fourier Transform (FT), f is decomposed into sinusoids (sine and cosine), whereas in the Wavelet Transforms (WT) it is decomposed into wavelets (*e.g.* Haar and Daubechies). FT produces the frequencies found in a signal, however, the spatial (time) information is suppressed in the Fourier domain. On the other hand, WT keeps both the frequency and time information of the signal. For this reason, a 1-D signal in wavelet domain is a 2-D function. In a single scale, WT decomposes the signal into high and low frequency components. In multi-scale decomposition, the low frequency part is recursively decomposed into high and low frequency components. Multi-scale wavelet decomposition is performed by defining a wavelet function ψ_0. By dilating and translating this *mother* wavelet function, several other wavelet functions can be generated.

Since WT approximates the signal with a high sparsity, it is useful to detect noise components in the signal and remove them. This can be performed by transforming the image into the wavelet domain, thresholding the noise coefficients and, then, applying the inverse wavelet transform [CYV00]. Soft thresholding leads to a less severe distortion of the signal of interest, therefore it is often the choice. The result of applying the wavelets soft thresholding to an image is shown in Figure 2.6f.

4.3.2 Curvelets

A curvelet is a multi-scale, directional, and parabolic scaled waveform [CD00]. A 2-D parametric curvelet function $\phi_{j,l,\boldsymbol{k}}(\boldsymbol{t})$ supported by a rectangle of length $2^{\frac{-j}{2}}$ and width 2^{-j} is defined as:

$$\phi_{j,l,\boldsymbol{k}}(\boldsymbol{t}) = 2^{\frac{3j}{4}} \phi(\mathcal{D}_j \mathcal{R}_{\theta_{j,l}} \boldsymbol{t} - \boldsymbol{k}) \tag{2.16}$$

where $j \geq 0$, $0 \leq l \leq 2^{-\lfloor \frac{j}{2} \rfloor} - 1$, and $\boldsymbol{k} \in \mathbb{Z}^2$ are the dilation, orientation and translation parameters, respectively. $\lfloor \cdot \rfloor$ is the floor operator. $\boldsymbol{t}$ is the time domain variable, and $2^{\frac{3j}{4}}$ is a normalization scaler.

$$\mathcal{D}_j = \begin{bmatrix} 2^{-j} & 0 \\ 0 & 2^{\frac{-j}{2}} \end{bmatrix} \tag{2.17}$$

is the parabolic dilation matrix, and

$$\mathcal{R}_{\theta_{j,l}} = \begin{bmatrix} \cos\theta_{j,l} & \sin\theta_{j,l} \\ -\sin\theta_{j,l} & \cos\theta_{j,l} \end{bmatrix} \quad , \ \theta_{j,l} = 2\pi l 2^{-\lfloor \frac{j}{2} \rfloor} \tag{2.18}$$

is the scale-dependent rotation matrix.

In order to generate a curvelet function, a mother curvelet function is required. In the Fourier domain, a mother curvelet function is defined by the support of a wedge created by the product of two window functions: W (radial), and V (angular):

$$\hat{\phi}_{j,0,0}(r,\omega) = 2^{\frac{-3}{4}} W(2^{-j}r) V \left(\frac{2^{\lfloor \frac{j}{2} \rfloor}\omega}{2\pi} \right) \tag{2.19}$$

where $r = \sqrt{\zeta_1^2 + \zeta_2^2}$ and $\omega = \tan^{-1}\left(\frac{\zeta_1}{\zeta_2}\right)$ are, respectively, the radius and angle of the polar coordinates of the frequency domain, and $\boldsymbol{\zeta} = [\zeta_1 \ \zeta_2]$ is the frequency domain variable. While W is responsible for defining the scale, V is responsible for defining the orientation, and they must satisfy some admissibility conditions.

In the frequency domain, a curvelet parameterized by $j, \boldsymbol{k}, l$ is defined as:

$$\hat{\phi}_{j,l,\boldsymbol{k}} = \hat{\phi}_{j,0,0}(\mathcal{R}_{\theta_{j,l}}\boldsymbol{\zeta}) e^{i\langle \boldsymbol{k}^{(j,l)}, \boldsymbol{\zeta} \rangle} \tag{2.20}$$

where $\boldsymbol{k}^{(j,l)} = \mathcal{R}^{-1}_{\theta_{j,l}} \begin{bmatrix} k_1 2^{-j} \\ k_2 2^{\frac{-j}{2}} \end{bmatrix}$.

The curvelet transform of a function $f \in L^2(\mathbb{R}^2)$, where L^2 is the space of square integrable functions, is defined as the inner product between a curvelet and f, $\langle f, \phi_{j,l,\boldsymbol{k}} \rangle$, and satisfies the Parseval's identity:

$$||f||_2^2 = \sum_{j,l,\boldsymbol{k}} |\langle f, \phi_{j,l,\boldsymbol{k}} \rangle|^2 \tag{2.21}$$

and the reconstruction of f is given by:

$$f = \sum_{j,l,\boldsymbol{k}} \langle f, \phi_{j,l,\boldsymbol{k}} \rangle \phi_{j,l,\boldsymbol{k}} \tag{2.22}$$

Similarly to the wavelet transform, the curvelet transform decomposes a function f into a set of weighed basis functions $\phi_{j,l,\boldsymbol{k}}$. Noise can be suppressed from an image using curvelets by coefficient thresholding (Figure 2.6e). In fact, since curvelets are multi-scale oriented waveforms, when compared to wavelets, they are more suitable to handle curvilinear structures, because approximating an edge will require many wavelet functions and, a single curvelet may be sufficient to model it.

For discretization and implementation, the warping method implemented in the *Curvelab* package is used to calculate the discrete curvelet transform. In Section 3 in Chapter 7, curvelets are used to develop a method for rings' suppression in CT volumes.

5 Image Segmentation

Image segmentation is the process of dividing the input image into a set of disjoint regions. Each region corresponds to a single object. In many cases, binary segmentation is sufficient and the main goal is to produce a binary image. This section briefly addresses common existing segmentation methods.

In order to choose a segmentation method that suites the input images, there are many factors to be considered, such as:

- Number of classes in the input image;
- Smoothness and accuracy required for objects' contours;
- Contrast and illumination variation;
- Computation demands;
- Level of user interaction;

- Signal-to-noise ratio (SNR);
- Traits of the objects (shape priors).

In the simplest case scenario, the input image contains two classes with good SNR, good contrast, and uniform illumination. In this case, *thresholding* is perhaps the most common approach. When more classes are defined, *clustering* can be used to obtain multi-class segmentation. In addition, when initial seeds can be provided by the user, *region growing* is recommended. When tight contours are needed, and initial contours can be provided by the user, *active contours* is a good option. Moreover, methods like graph-cut and grow-cut are often used in medical applications and image editing. For segmenting images of simple shapes, Mathematical Morphology (MM) approaches are efficient, among them, the popular Watershed Transform (WST) is shown to be very powerful approach. This transform is described in Section 3.2 in Chapter 3.

5.1 Thresholding

Perhaps, the best way to define image segmentation by thresholding is to borrow Otsu's definition [Ots75]. Given an image histogram, image thresholding consists in seeking a threshold that separates the histogram into two classes, such that, the variation between the classes (inter-class variance) is maximum and the variation within each class (intra-class variance) is minimum. However, this definition assumes a histogram of clearly separable classes (bi-modal). Assuming a dark foreground on a light background, the thresholding function is defined as:

$$\boldsymbol{S}^{(\text{binary})} = \begin{cases} \text{foreground}, & \boldsymbol{S} < \tau \\ \text{background}, & \text{otherwise} \end{cases} \tag{2.23}$$

where $\boldsymbol{S}^{(\text{binary})}$ is the binary image and τ is the threshold. Thresholding can be global or local. In global thresholding, a single threshold (or a few number of thresholds) is used to binarize the whole image. In the latter, each region in the image is assigned a threshold.

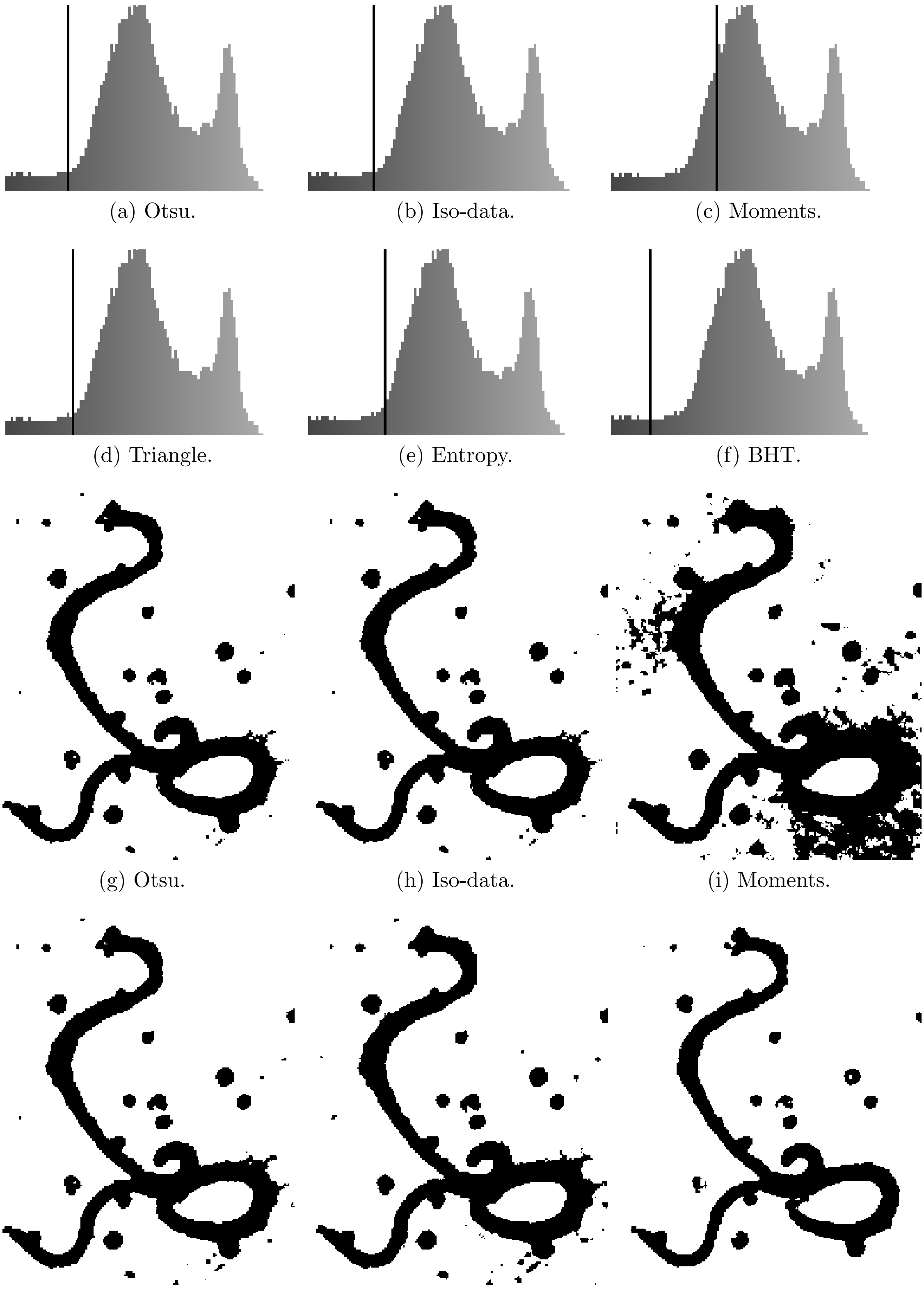
5.1.1 Global Thresholding

In addition to Otsu's method, there are many other thresholding approaches that can be applied to the histogram, the interested reader is refereed to [SS04]. Up to our knowledge, the Iso-data [RC78] and Balanced Histogram Thresholding (BHT) [dAS08] approaches perform nicely in many cases. Examples of some methods are shown in Figure 2.7.

5.1.2 Local Thresholding

A single thresholding is not efficient under uneven illumination, therefore, applying local thresholding is more robust than looking for a global threshold, since the illumination is assumed to be a slow varying field that is locally uniform (Chapter 3). In addition, since illumination and contrast are difficult to interpret in a global histogram, in local thresholding the input image is divided into a set of overlapping regions, and each region is processed independently. Inside each region, a statistical measure (*e.g.* the mean) is used as a threshold for the region. An example of local thresholding is the mean-c approach [PW96].

In this work, mean-c is efficiently implemented by means of the integral image [VJ04] (Appendix A). In Chapter 6, this method is used to generate microfilariae candidates. In mean-c, the mean of a window is calculated and then subtracted from an intercept c, and each pixel is thresholded with respect to the mean. Therefore, two parameters



(a) Otsu. (b) Iso-data. (c) Moments.

(d) Triangle. (e) Entropy. (f) BHT.

(g) Otsu. (h) Iso-data. (i) Moments.

(j) Triangle. (k) Entropy. (l) BHT.

Figure 2.7: Image thresholding using different approaches for the sample in Figure 2.1. Top: the calculated thresholds; and bottom: the corresponding binary images.

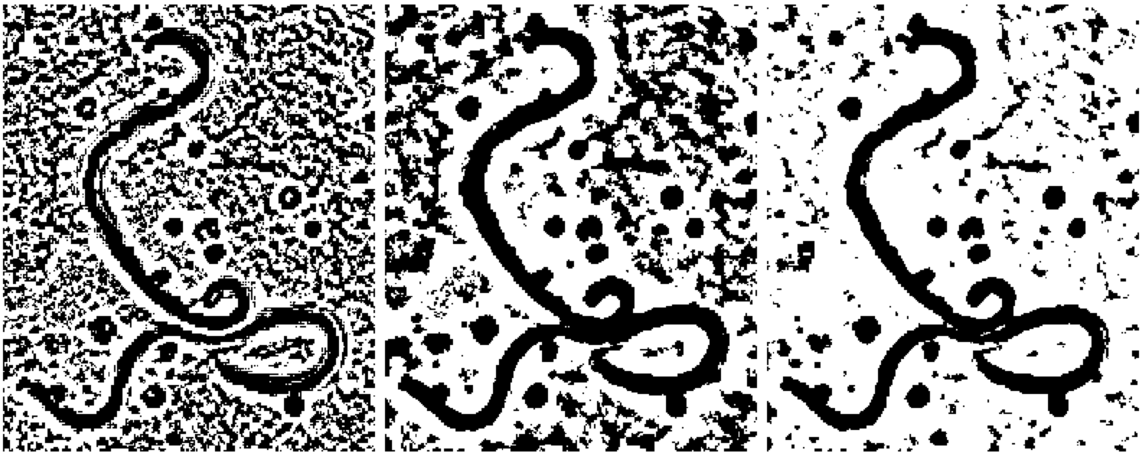
have to be set, the window size and c. Small windows may partially segment the objects (over-segmentation), whereas large windows may include background regions in the object. In practice, this parameter can easily be set. For instance, it has been found that only a slight change in the segmentation results is noticed in windows of radii 50 and 75 pixels, which means there is low sensitivity to rough estimations of this parameter. Parameter c defines the sensitivity of the segmentation to faint objects. High values ignore faint objects, and very low values will include background pixels. Therefore, a trade-off is needed to obtain good results. Setting this parameter within the range $[0.07, 0.15]$ of the window size is quite sufficient for the problem at hand, as depicted in Figure 2.8.

In this figure, when the window size is not sufficient, the microfilaria is partially segmented, and when c is small, mean-c detects background pixels as foreground. Once appropriate parameters are chosen, this method usually produces better results than global thresholding.

5.2 Edge Based Segmentation

One of the classical but still very helpful approach, besides basic thresholding, is edge detection. An edge in an image is a sudden change in the intensity. In other words, objects can be regarded as homogeneous regions separated by pronounced borders. In general, the factors that create edges are:

- Light discontinuity (*e.g.* object shadow);
- Color discontinuity;
- Discontinuity in the surface normal;



(a) Window size $= 10$ and $c = 0$. (b) Window size $= 30$ and $c = 0$. (c) Window size $= 30$ and $c = 5$.

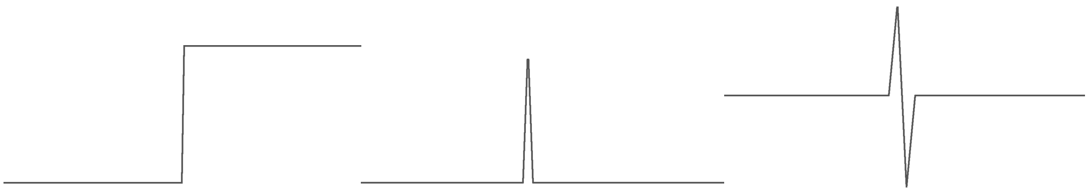
(d) Window size $= 130$ and $c = 15$.

Figure 2.8: The response of mean-c local thresholding method applied to the sample in Figure 2.1 with different parameters.

- Change in the depth, primarily due to obtaining a 2-D image from a 3-D object.

5.2.1 Image Derivatives

In order to detect edges, it is important to see how the gradient changes across the image domain. That is because the edges respond as pikes in the first order derivative and lie at the zero-crossing of the second order derivative, this is illustrated in Figure 2.9 for a clear step edge.



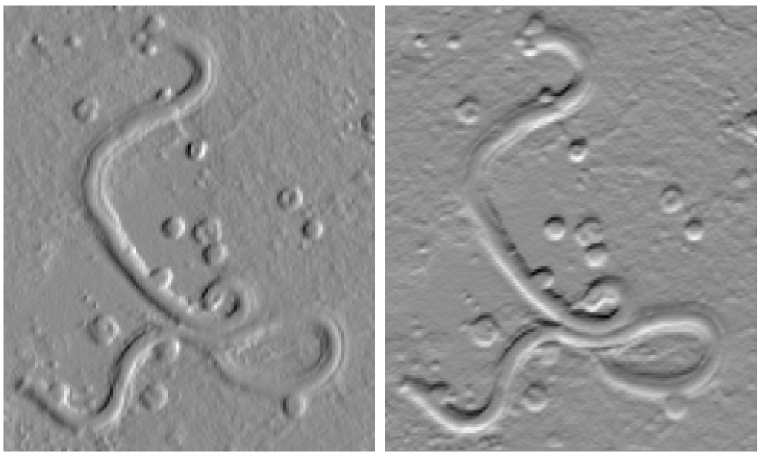
(a) A step edge. (b) 1[st] derivative. (c) 2[nd] derivative.

Figure 2.9: A typical step edge and its first and second order derivatives.

The image gradient $\nabla \boldsymbol{S}$ is a vector containing the image derivatives with respect to x and y:

$$\nabla \boldsymbol{S} = \begin{bmatrix} L_x \\ L_y \end{bmatrix} \tag{2.24}$$

where $L_x = \frac{\partial \boldsymbol{S}}{\partial x}$ and $L_y = \frac{\partial \boldsymbol{S}}{\partial y}$ are the directional first order derivatives in x and y, respectively. These are depicted in Figure 2.10.



(a) L_x. (b) L_y.

Figure 2.10: Directional derivatives of the image shown in Figure 2.1.

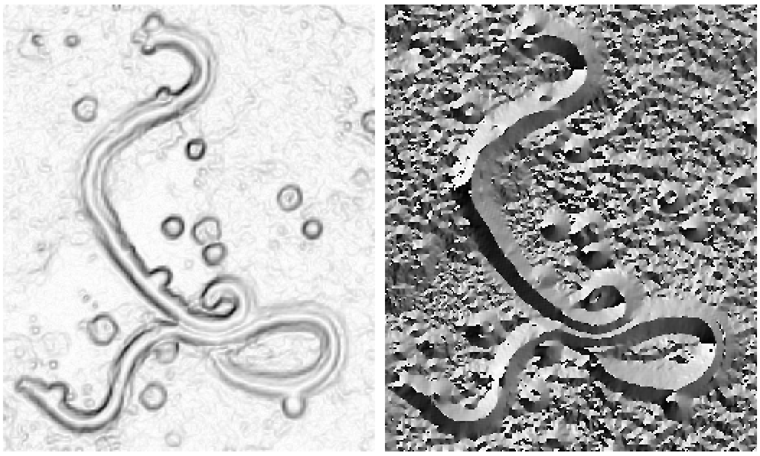
The gradient magnitude of $\boldsymbol{S}$ is:

$$|\nabla \boldsymbol{S}| = \sqrt{L_x^2 + L_y^2} \tag{2.25}$$

and the gradient orientation is $\theta_{\nabla \boldsymbol{S}}$:

$$\theta_{\nabla \boldsymbol{S}} = \tan^{-1}\left(\frac{L_y}{L_x}\right) \tag{2.26}$$

These are shown in Figure 2.11.



(a) Gradient magnitude. (b) Gradient orientation.

Figure 2.11: Gradient magnitude and orientation of the image shown in Figure 2.1.

Since the calculation of the image derivatives has to be carried out on a grid, *i.e.* a discrete domain, the derivatives need to be approximated. For this purpose, images can be convolved with the finite differences kernels:

$$L_x \approx \boldsymbol{S} * \begin{bmatrix} -1 & 0 & +1 \end{bmatrix} \tag{2.27}$$

and

$$L_y \approx \boldsymbol{S} * \begin{bmatrix} -1 \\ 0 \\ +1 \end{bmatrix} \tag{2.28}$$

Another common approximation is to replace the finite differences kernels by the Sobel–Feldman operators:

$$L_x \approx \boldsymbol{S} * \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{2.29}$$

and similarly for L_y:

$$L_y \approx \boldsymbol{S} * \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{2.30}$$

The derivatives in Figure 2.10 were calculated using these operators.

The second order derivatives can be approximated as:

$$L_{xx} \approx \boldsymbol{S} * \begin{bmatrix} +1 & -2 & +1 \end{bmatrix} \tag{2.31}$$

and likewise for y:

$$L_{yy} \approx \boldsymbol{S} * \begin{bmatrix} +1 \\ -2 \\ +1 \end{bmatrix} \tag{2.32}$$

L_{xy} is calculated as:

$$L_{xy} \approx \boldsymbol{S} * \begin{bmatrix} -0.25 & 0 & 0.25 \\ 0 & 0 & 0 \\ 0.25 & 0 & -0.25 \end{bmatrix} \tag{2.33}$$

5.2.1.1 Laplacian of Gaussian The Laplacian operator $\Delta \boldsymbol{S}$ is defined as:

$$\Delta \boldsymbol{S} = L_{xx} + L_{yy} \tag{2.34}$$

The image Laplacian can be approximated by convolving the image with the kernel:

$$\Delta \boldsymbol{S} \approx \boldsymbol{S} * \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & -1 \end{bmatrix} \tag{2.35}$$

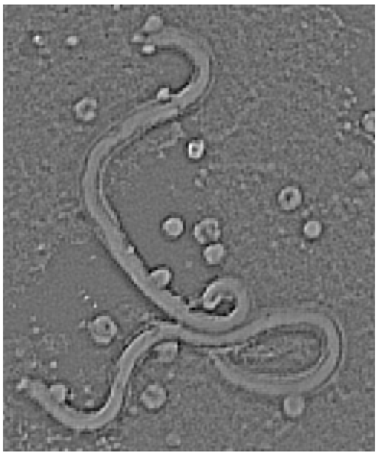
Since second order derivatives are sensitive to noise, they are preceded by a low-pass filtering. When applying the Gaussian kernel to suppress noise, the resulting operation is called Laplacian of Gaussian (LoG). The LoG function is defined as:

$$\mathrm{LoG}(\boldsymbol{S}) = -\frac{1}{\pi\sigma^2}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.36}$$

and the result of finding the zero-crossing of the LoG is shown in Figure 2.13b.

In fact, LoG works as a matched filter (Figure 2.12). Whenever the size of the kernel fits the "blob", the response becomes higher, which is a helpful property to detect blobs. It will be shown in Chapters 4, 5, and 6 that LoG is a powerful filter to design versatile object detectors. Additionally, LoG can be approximated by the Difference of Gaussians (DoG) [Low04], which is the difference between two images filtered by a Gaussian with different standard deviations.

5.2.1.2 Gaussian Derivatives Perhaps, one of the most important kernels in image analysis is the Gaussian kernel. In Chapters 4, 5, and 6, it is shown that, this kernel plays a major role in calculating image's features. When this kernel is used, it becomes easy to design scale-space methods (Section 2.1 of Chapter 6). Compared



(a) $\sigma = 0.5$.

(b) $\sigma = 2$.

(c) $\sigma = 5$.

(d) $\sigma = 10$.

Figure 2.12: The LoG filter with different σ applied to the sample in Figure 2.1.

to the finite differences approximation of the derivatives, Gaussian image derivatives are more robust to noise [Ste98].

In order to calculate the Gaussian image derivatives, the image is convolved with the derivatives of the 2-D Gaussian kernel $\mathcal{G}_\sigma(x, y)$. The first order derivatives of the Gaussian are calculated as:

$$\frac{\partial}{\partial x}\mathcal{G}_\sigma(x, y) = -\frac{x}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.37}$$

$$\frac{\partial}{\partial y}\mathcal{G}_\sigma(x,y) = -\frac{y}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.38}$$

and the second order derivatives are calculated as:

$$\frac{\partial^2}{\partial^2 x}\mathcal{G}_\sigma(x,y) = \frac{\frac{x^2}{\sigma^2}-1}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.39}$$

$$\frac{\partial^2}{\partial^2 y}\mathcal{G}_\sigma(x,y) = \frac{\frac{y^2}{\sigma^2}-1}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.40}$$

$$\frac{\partial^2}{\partial xy}\mathcal{G}_\sigma(x,y) = \frac{xy}{2\pi\sigma^6}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.41}$$

In fact, edge detection has been very important not only in segmentation problems, but also in many other problems in computer vision like object description. Among edge detection methods, the method developed by John Canny [Can86] (Figure 2.13c) is one of the most popular. It is a multi-stage method where the noise is reduced by a low-pass Gaussian filter, followed by the computation of the gradient magnitude (Equation 2.25). The non-maximum suppression is then applied to obtain single-lined edges. This is an important property, since for any post processing task, it is helpful to deal with a single edge rather than a "blurred edge" (Figure 2.11). In order to get rid of weak edges, hysteresis thresholding is used. A state-of-the-art edge detector was designed by Pablo Arbelaéz *et* al. [AMFM11], where several cues are combined to develop a powerful edge detector.

In Figure 2.13, Sobel, LoG, and Canny edge detectors were compared. It is clear that, Canny's method is preferable due its ability to produce better results for post processing tasks such as edge linking. However, the Canny's low-pass filtering might shift the edge position.

(a) Sobel. (b) Zero-crossing of LoG. (c) Canny.

Figure 2.13: Edge detection methods for the sample in Figure 2.1.

5.3 Clustering

Clustering can be seen as an extension to thresholding. Both are pixel based segmentation methods, but in clustering, a higher dimension feature space can be used instead of a single histogram. The aim of clustering is to perform unsupervised assignment of a pixel to a cluster given the pixel's feature vector. Features can be a single value (*e.g.* a pixel gray level intensity), multiple values such as pixel's RGB values, or a vector calculated from the pixel's neighborhood. An example of image segmentation by clustering in the Hue-Saturation-Value (HSV) space using the k-means method is shown in Figure 2.14.

In some applications, clustering is used as initial segmentation, where region growing with Markov Random Field (MRF) is then used to refine the segmentation. MRFs are random fields that satisfy the Markov's property. In essence, this property states that, the value of a pixel is restricted by its neighbors. This is a very useful property

(a) 4 Classes.

(b) 3 Classes.

(c) 2 Classes.

Figure 2.14: Image segmentation by clustering using different number of classes in the HSV space.

for image segmentation to cope with noise and small holes in piecewise smooth regions [ENM+14].

5.4 Graph Cut

When using graph theory, an image can be thought of as a weighed graph, where pixels serve as vertices (nodes) and the similarities between them act as weighed edges. The idea is to separate the image domain, by a cut, given two user defined nodes: the source (foreground) and the sink (background) [BVZ01]. This is an optimization problem where the objective is to assign each pixel to either the source or the sink via finding a decision boundary (the minimum cut), without violating the segmentation constraint. This constraint requires each vertex to be connected either to the source or the sink, but not to both. This method is basically divided into two main steps: generating the graph, and finding the minimum cut.

In order to generate the graph, all vertices are initially connected to the source and the sink. The edges are calculated with respect to the probability that a vertex belongs to the source or the sink. The neighbor vertices are also connected, such that the weights of the links between them encourage similar vertices to stay together after segmentation. Generating these links is an application domain problem, where several criteria can be used to calculate the weights. The minimum cut consists of removing a set of edges, in such a way that, the accumulated sum of edges' weights is minimum and the segmentation constraint is fulfilled.

5.5 Region Growing

In region filling, the color (or intensity) is propagated (flooded) based on a given neighborhood connectivity (*e.g.* 8-connected). This is, basically, the main idea behind region growing segmentation. Generally, one needs to set the initial seeds, and the growth criteria [AB94]. In addition, post-processing might be needed to cope with noise and small unwanted holes. Initial seeds can be pixels, or regions (marks), and the flooding criteria considers:

- The proximity between the seed and neighbor pixels;
- The similarity between the seed and the neighbor pixels.

Further criteria can be considered such as the size (or shape) of the generated regions.

Seed detection can be automatic. For instance [SC05] used the YCbCr color space to detect seeds in color images. The seeds are detected based on the similarity between the seed and its neighbors in a 3×3 window. Manual seeds can also be provided, with interactive corrections as in [VK05].

Region growing can also be regarded as a competition between regions. For instance, inspired from the cellular automata, [VK05] designed a region growing method (grow-cut) where seeds act as bacteria competing for the domination of the whole image. In Figure 2.15, this method is seeded by the clustering results of 3 classes, and it was able to converge to reasonable results starting from good seeds.

(a) Seeds (3 clusters). (b) Binary result.

Figure 2.15: Results of applying grow-cut method [VK05] on the sample in Figure 2.1.

5.6 Active Contours

An alternative to calculating the edges or the binary image, a curve evolution process can be used to generate contours. A closed curve (contour) moving under its curvature will evolve into a circle and then vanish. Starting from a predefined contour, and allowing the curve to move until some condition is met, makes it is possible to create contours that fit objects' boundaries. This is the main idea behind active contours. They were originally proposed by Kass *et* al. [KWT88], where they were called *snakes*. Since then, several approaches have been developed. Among them, perhaps, the *level set* approaches [CV01] are attracting the researches' attention more than the others.

A 2-D grayscale image can be thought of as a bi-variate function (see Section 2), and the image intensity can be regarded as the surface plot of the image grid. This yields the triplet (x, y, l), where l is the intensity and is represented in the z-axis in Figure 2.16. A level curve can be seen as the contour of the function at a given point. In other words, if inserting a plane, such that it horizontally bisects the intensity at a given level, the resulting intersection between the plane and the intensity will be a level contour. For instance, consider a 2-D Gaussian function as in Figure 2.17, when placing a rectangular plane at the top of the Gaussian, a small circle will emerge at the intersection between the Gaussian and the plane. The radius of this circle grows as the plane moves towards the bottom of the Gaussian.

Some of the level set methods [CV01] are based on the Mumford-Shah image model [MS89]:

$$J_{\text{Mumford-Shah}}(\boldsymbol{S}; \alpha, \beta) = \alpha \int_{\Omega} (\boldsymbol{S} - \hat{\boldsymbol{S}})^2 + \int_{\Omega/C} |\nabla \hat{\boldsymbol{S}}| + \beta |C| \tag{2.42}$$

where α and β are user defined parameters, $\hat{\boldsymbol{S}}$ is the estimated image and C is the set of contours of the segmented regions.

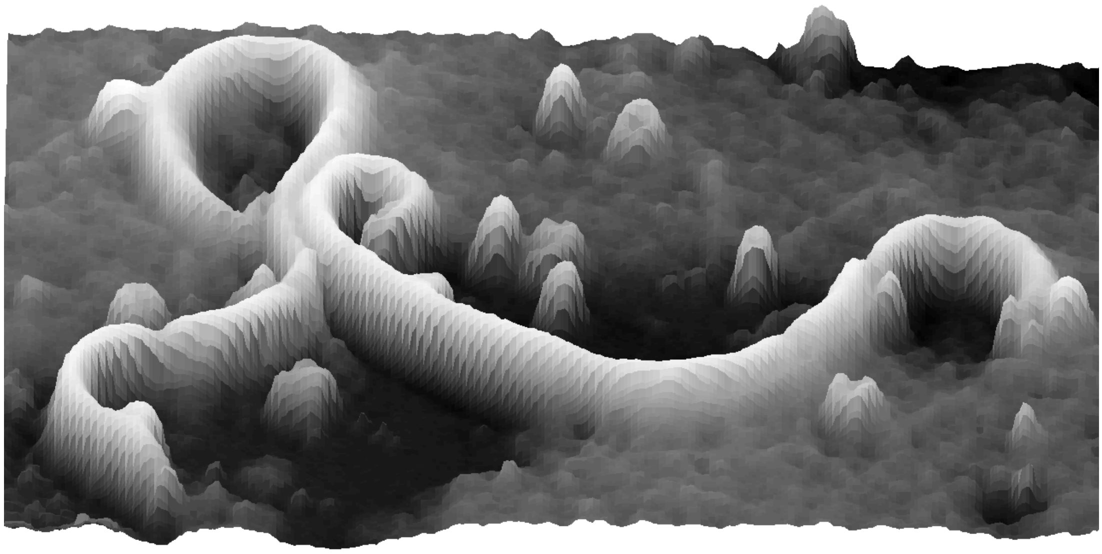


Figure 2.16: The surface plot of the sample in Figure 2.1, where the intensity is inverted for a better visualization.

In Equation 2.42, the first term is the fidelity term. The second term encourages the estimated image to be smooth except at the boundaries. The last term penalizes the length of the contour to ensure tight contours. By minimizing this functional, the resulting image is a cartoon version of the original (*i.e.* each region is constant with sudden changes at its boundaries). The level set without edges (aka Chan-Vese model) is a special case of the Mumford-Shah model. This method is used to generate the results in Figure 2.18. In this figure, although level set's segmentation produces fine contours, the initial curves need to be placed near to the object boundaries, otherwise, it might stop at a non-ideal solution. This can be very useful to obtain accurate contours after a crude initial estimation, as in medical image analysis.

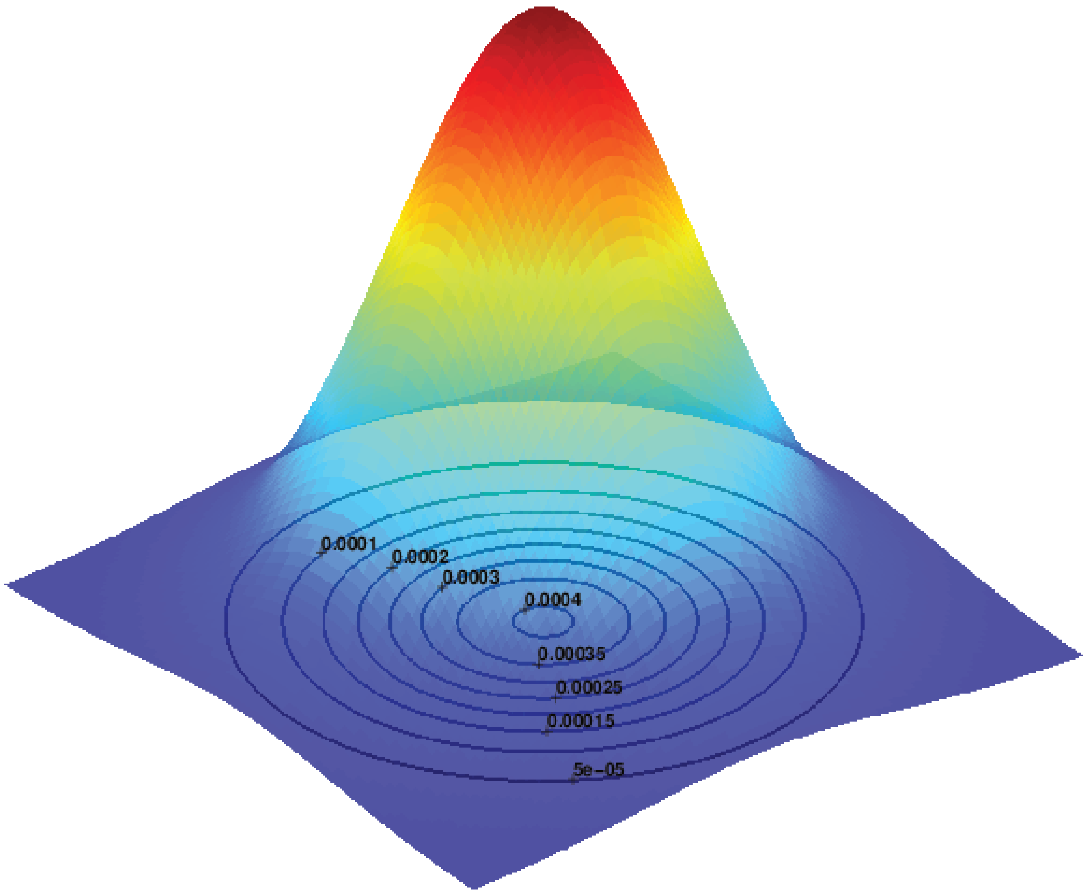
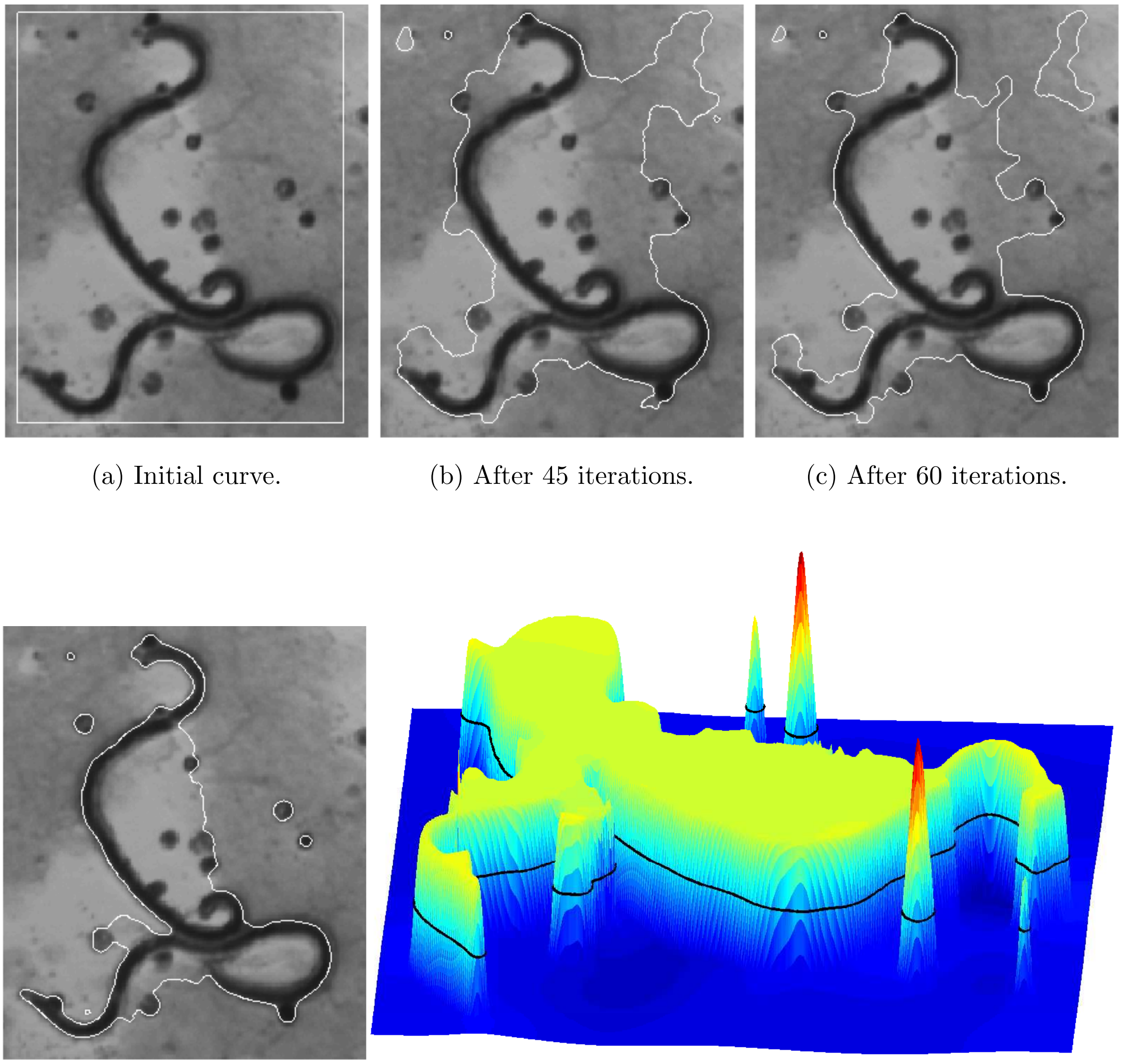


Figure 2.17: A 2-D Gaussian function and its curve levels at some points.

6 The Hessian

An image can be approximated at the neighborhood of a point $\boldsymbol{p}$ by using the Taylor expansion [FNVV98]:

$$\boldsymbol{S}_{\boldsymbol{p}+k\boldsymbol{p}} \approx \boldsymbol{S}_{\boldsymbol{p}} + k\boldsymbol{p}^{\top} \nabla \boldsymbol{S}_{\boldsymbol{p}} + k\boldsymbol{p}^{\top} \mathcal{H}_{\boldsymbol{p}} k\boldsymbol{p} \tag{2.43}$$



(a) Initial curve. (b) After 45 iterations. (c) After 60 iterations.

(d) After 300 iterations. (e) Final level curve (black line).

Figure 2.18: Level set active contour segmentation of the sample in Figure 2.1.

where $\boldsymbol{S_p}$ is the intensity of image $\boldsymbol{S}$ at $\boldsymbol{p}$ and k is a scaling factor. $\mathcal{H}$ is the Hessian, which is a symmetric matrix calculated by using the second order derivatives of $\boldsymbol{S}$:

$$\mathcal{H} = \begin{bmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{bmatrix} \tag{2.44}$$

Since $\mathcal{H}$ describes how the normal to an isosurface changes [HKG00], it is useful for shape encoding. Therefore, $\mathcal{H}$ is important and will be used frequently in Chapters

4, 5, and 6. The *eigenpairs* $\{\boldsymbol{v_1}, \lambda_1\}$ and $\{\boldsymbol{v_2}, \lambda_2\}$ of this matrix are the solution of the *eigensystem*:

$$\mathcal{H}\boldsymbol{v} = \lambda\boldsymbol{v} \tag{2.45}$$

where λ is the eigenvalue associated with the eigenvector $\boldsymbol{v}$ of the eigenpair $\{\boldsymbol{v}, \lambda\}$. In this case, $\mathcal{H}$ is calculated with respect to x and y, therefore resulting in two eigenpairs. These are related to $\mathcal{H}$ by:

$$\mathcal{H} = \boldsymbol{V}\mathrm{diag}(\lambda_1, \lambda_2)\boldsymbol{V}^{-1} \tag{2.46}$$

where the matrix $\boldsymbol{V} = [\boldsymbol{v_1} \quad \boldsymbol{v_2}]$, and $\mathrm{diag}(\lambda_1, \lambda_2)$ is the diagonal matrix of the eigenvalues. Additionally, the Laplacian (Equation 2.34) is related to $\mathcal{H}$ by:

$$\Delta\boldsymbol{S} = \mathrm{Tr}(\mathcal{H}) \tag{2.47}$$

where Tr is the trace of the matrix.

Eigenanalysis of the Hessian can be used to find the principal components (orthogonal eigenvectors) of a thin object at a given point. One of the eigenvectors points to the largest curvature and the other points to the direction of the structure. In the *absolute ordering* [FNVV98], where the eigenvalues are ordered according to their absolute values, the eigenvector perpendicular to the structure is associated to the highest eigenvalue, while the other vector is associated to the lowest eigenvalue. This will be used in the proposed approach in Chapter 6 to find the local orientation of the microfilariae.

7 Connected Component Analysis

In this section Connected Components (CC), Distance Map (DM), and Medial Axis Transform (MAT) are discussed. These are essential concepts in binary image processing.

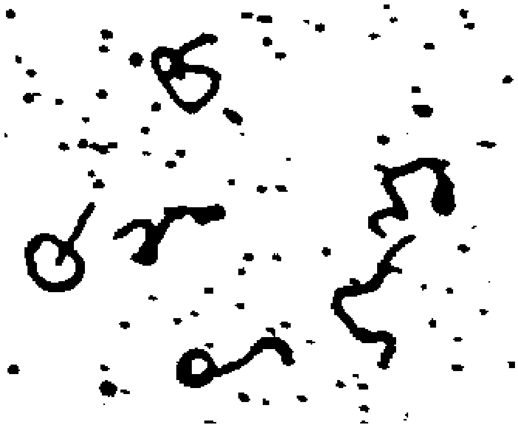
7.1 Connected Components

A connected component is a set of neighboring pixels with the same label. A CC of size k is denoted by the set of pixels $P = \{\boldsymbol{p}_1, \cdots \boldsymbol{p}_k\}$, where $\boldsymbol{p} = [x \quad y]^\top$ is a point in the 2-D image grid. CC analysis is meant to study the object's geometric features such as eccentricity, compactness, and so on. The interested reader is refereed to Chapter 11 in [BB09]. In order to analyze objects in a binary image, they must be identified beforehand. This process is called Connected Components Labeling (CCL), where each object is assigned a label (an integer number). An example is shown in Figure 2.19.

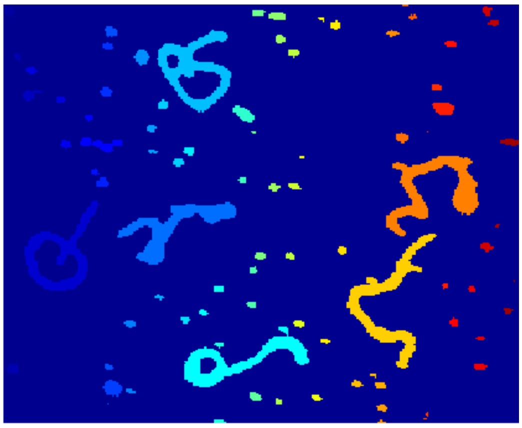
7.2 Distance Map

A Distance Map (DM) is a function that measures the distance of an object's pixel to its nearest background pixel. There are several distance measures that can be used, such as Euclidean (ℓ_2), Manhattan (ℓ_1), and so on. In this thesis, the Euclidean Distance Map (EDM) is used. There are many ways to calculate the EDM, such as the Meijster's approach [MRH02], which is found to be efficient in terms of computation. DM is very helpful for many applications, especially in morphological image analysis. EDM is used in Chapter 6 to compute the local width of the microfilariae. An example of the EDM is depicted in Figure 2.20.

(a) A microfilaria image.



(b) Connected components.



(c) Color encoded labels.

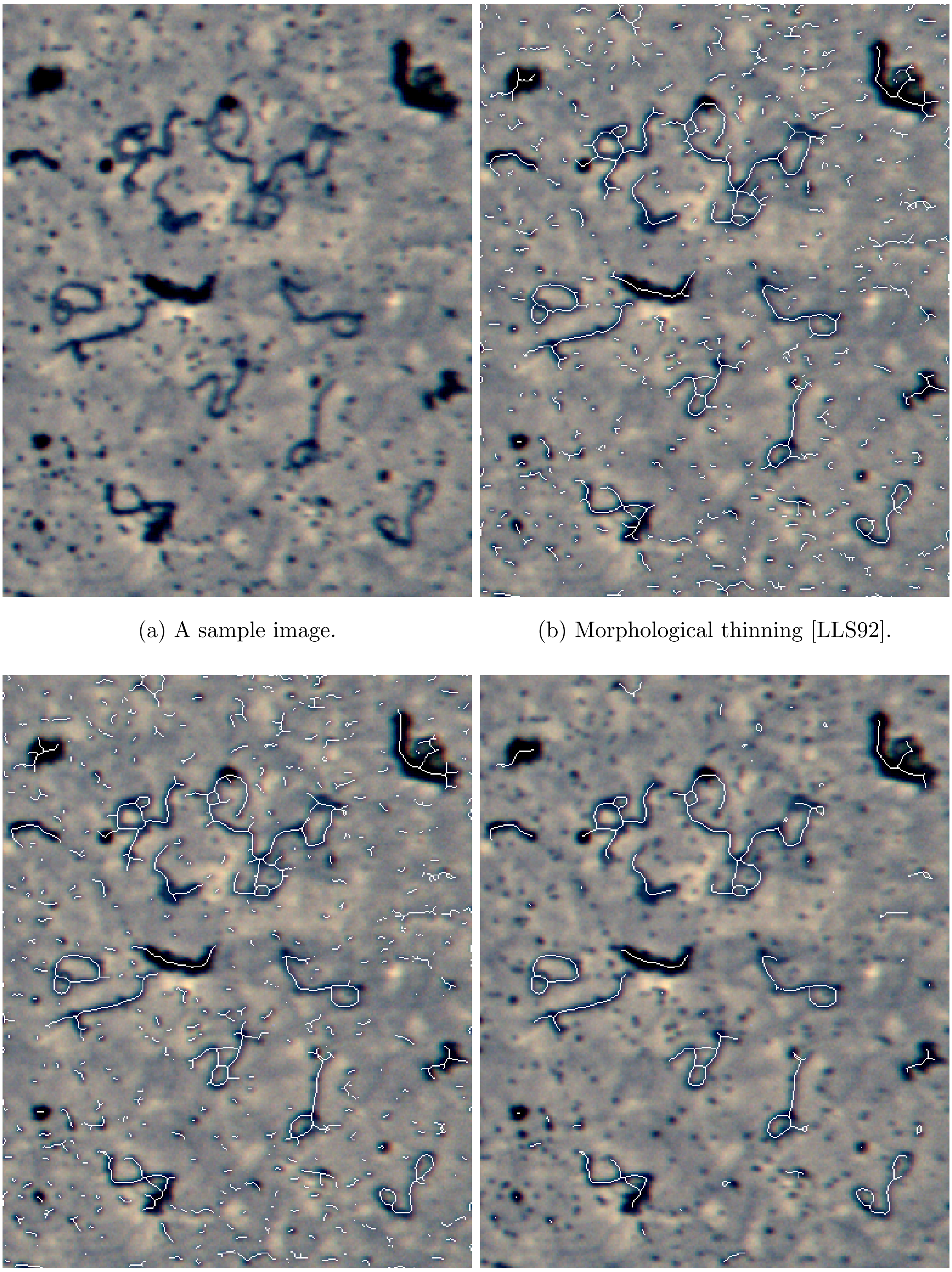
Figure 2.19: CCL for a sample microfilaria image.

Figure 2.20: EDM of the binary image in Figure 2.8d. The darker the higher the distance.

7.3 Medial Axis Transform

Medial Axis (MA) is an important feature of binary objects, because when the DM is calculated precisely at each MA point, the binary object can be recreated from its MA and DM. The reason for that is because, a point in the MA is the center of the maximal disk that can fit inside the object without crossing the object's borders. In fact, MA and DM have been used in the Quench function concept, for binary image compression [Vin00]. A process called *thinning* is used to calculate the MA. It can be thought of as fire fronts emerging from the binary object's boundaries and stopping at the center of the object.

A well-known problem of thinning is, however, the MA's sensitivity to contour's irregularities. Therefore, it is important to rely on a good thinning approach. In the literature, it is a common practice to obtain a rough skeleton using fast methods, *e.g.* Morphological Thinning (MT) [LLS92] or Zhang and Suen (ZS) [ZS84] and, then, applying a pruning strategy, such as, branch length threshold or discrete curve evolution (DCE) [BLL07]. Although MT and ZS produce spiky skeletons, they can be implemented efficiently. In what concerns the problem at hand, ZS is producing the best results. In this work, a thinning approach presented by Telea in [TVW02] is used. In Figure 2.21, Telea shows superiority compared to the other two "classical" methods, however, the main drawback of this method is that it is a little bit slower.



(a) A sample image.

(b) Morphological thinning [LLS92].

(c) Zhang and Suen thinning [ZS84].

(d) Telea thinning [TVW02].

Figure 2.21: Medial axis detected by various approaches superimposed on a sample. These approaches were applied on a binary version obtained by the local-mean approach.

3

ILLUMINATION CORRECTION

1 Introduction

IMAGE restoration is an ill-posed inverse problem of recovering the "original" image from the degraded one [GW02]. Illumination correction is one of the fundamental problems in image restoration. Uneven illumination is a recurrent problem in microscopic and medical images. During image acquisition and formation, the quality of the device and the expertise of the operator play an important role in obtaining high quality images. Unfortunately, neither of these two requirements is always fulfilled. The shading resulting from poor and/or uneven illumination has direct impact on the complexity of both manual and automatic image analysis processes. It is very important that an image is uniformly illuminated so that, particles of the same class, have the same brightness regardless of their positions. The lack of this criterion makes detection and analysis of objects an extremely difficult task [TLP02].

This chapter is organized as follows: Section 2 briefly describes the work related to illumination bias estimation. The proposed approach is described in Section 3. Finally, the results and the conclusion are presented in Sections 4 and 5, respectively.

2 Related Work

Although most of the literature focuses on correcting the illumination bias field in Magnetic Resonance Imaging (MRI) images of the brain [VPL07], the taxonomy of illumination correction methods can be organized into two main groups: prospective and retrospective. A common practice in the prospective approach is to obtain a "blank" image (without any object) merely showing the bias field. This reference image is then used to minimize the illumination problem in images produced by the same device and using the same parameters. This approach is not practical because the parameters are usually sample dependent and, most of the times, only the sample images are available, making it hard to retrieve the "real background". On the other hand, the retrospective approach is data-driven, being more functional as the bias field can be estimated from the input image directly.

In Equation 2.1, the observed image is modeled as a multiplicative model, where $\boldsymbol{B}$ represents a multiplicative illumination field. Since noise suppression may be dealt with independently, as shown in Section 4 of Chapter 2, the noise term $\boldsymbol{\epsilon}$ can be ignored, and Equation 2.1 can be rewritten as:

$$\log(\boldsymbol{S}) = \log(\boldsymbol{T}) + \log(\boldsymbol{B}) \tag{3.1}$$

Once $\boldsymbol{B}$ is calculated, $\boldsymbol{T}$ can easily be obtained. Nevertheless, finding $\boldsymbol{B}$ is an ill-posed problem, since in Equation 3.1 there are two unknown variables $\boldsymbol{T}$ and $\boldsymbol{B}$. However, information about both $\boldsymbol{T}$ and $\boldsymbol{B}$, can be used to constrain their estimations and convert Equation 3.1 into a well-posed problem.

$\boldsymbol{B}$ is assumed to be a smooth varying intensity field, having low gradient magnitude. Therefore, lowpass filtering and mathematical morphology can be used to estimate $\boldsymbol{B}$. Although these approaches are efficient in terms of computation, they are not efficient when the foreground dominates the image. An improved lowpass filtering method, by remapping the intensity distribution and using bandpass filtering to preserve medium-sized objects, is described in [HM14].

The most common approach is to estimate $\boldsymbol{B}$ by using the support of a smooth 2-D parametric function, such as polynomials [ZG10, TJW08, ATdABS15], B-Splines [KJK10], and thin plate splines (TPS) [BSP04, AGS+12]. Besides being robust to noise, they also result in smooth fields. Nevertheless, estimating $\boldsymbol{B}$ parametrically presents a challenge. Complex bias fields need high order polynomials, but using polynomials of very high order may result in a model of the bias field that is highly disturbed by the objects. Therefore, there is a relation between the degree of the polynomial and the complexity of the bias in order to achieve good results. Although the parametric representation of $\boldsymbol{B}$ is compact, that is, the image is represented by a small set of parameters, a non-parametric estimation of $\boldsymbol{B}$ is also possible [ZGAG09].

Several preconditions can be used for $\boldsymbol{T}$ depending on the method. In [SZE98], $\boldsymbol{T}$ is perceived as a high frequency content in the image intensity distribution. The objective, in this case, is to find the bias field that maximizes this content. In [ZGAG09], $\boldsymbol{T}$ is represented by the sparsity principle [OF96], whereas the total variation (TV) (see Section 4.2 of Chapter 2) is used as a sparsity measure. In [TJW08, CYZ+05, MMOC11], the gradient of the image is used to constrain the solution of Equation 3.1 regarding $\boldsymbol{T}$. Entropy-based minimization is used in [KJK10, LMVP00]. Gradient magnitude is used in [SWKJ02, TJW08] to fit a 2-D polynomial to the corresponding intensities in the input image. In [SWKJ02], anisotropic diffusion is used to suppress the noise and then, a thresholding approach is applied to remove high magnitude pixels. The threshold of the gradient poses another problem. The magnitude of the under-illuminated regions in the image is low, even if they correspond to edges of objects. In [TJW08], anisotropic diffusion is replaced by Gaussian filtering, and a weighing approach is used instead of thresholding.

The emphasis on relying on the background can be observed in many papers. For instance, in [LB01], the background is used to improve the segmentation and bias correction interchangeably. In [LOD12], vascular and foveal pixels are extracted from the input image in order to retain the background and, then, a heat propagation image in-painting is applied to interpolate the missing pixels in the background in order to build a smooth illumination field. In other approaches, such as in [LGWG09], the segmentation and bias estimation are combined into a single model, which can be solved interchangeably by solving for the bias coefficient, and then updating the segmentation results. This depends on two important factors: The accuracy and stability of the bias estimation solution, and the type of image. The bias field in piecewise smooth images (*e.g.* MRI) has unique properties. For instance, it is assumed that noticeable bias fields decrease the sparsity of the gradient of MRI images [ZGAG09]. Furthermore, relatively large MRI regions are "constant". In microscopy images, these properties do not always hold [TJW08], except in the background. The use of segmentation has two major disadvantages: First, strong bias fields make it very difficult to come up with a good initial segmentation, resulting in methods converging to non-acceptable solutions. Second, it is very sample dependent.

Even when different approaches are used, all these methods somehow agree that the best pixels to estimate $\boldsymbol{B}$ reside in the image's background. This is due to the fact that the background is more likely to have the same intensity all over the image domain and, therefore, any smooth variation is in fact a consequence of the bias field. Moreover, the gradient behaves the same way in the background as it does in the bias field, except where noise is present. However, large piecewise smooth regions have similar properties to those in the background. In this work, this information is taken into account to help build a robust illumination correction approach.

3 The Proposed Approach

Given $\boldsymbol{S}$, according to Equation 3.1, the proposed method minimizes the amount of unwanted pixels used in estimating the bias field $\boldsymbol{B}$. It will be shown that the high gradient magnitude and/or foreground pixels can be avoided without the need for thresholding. For that purpose, two sampling solutions are suggested. The first one uses dynamic programming to avoid unwanted pixels. The second uses the watershed transform [VS91] to sample the background [dAME+11].

3.1 Sampling with Dynamic Programming (DP)

In this approach, the matrix $\boldsymbol{S}$ is divided into $k = 1, 2, 3, \cdots, \frac{n}{\alpha}$ vertical sections and into $l = 1, 2, 3, \cdots, \frac{m}{\beta}$ horizontal sections. Where α and β are the widths of the vertical and horizontal sections, respectively.

The kth vertical section is denoted by $\boldsymbol{S}^{(k)}_{m\times\alpha}$ and the lth horizontal section is denoted by $\left(\boldsymbol{S}^{\top}_{n\times\beta}\right)^{(l)}$, where $\top$ is the matrix transpose. This way, the kth vertical section is a submatrix defined as:

$$\boldsymbol{S}^{(k)}_{m\times\alpha} = \begin{bmatrix} s_{x_1,y_{\alpha(k-1)+1}} & s_{x_1,y_{\alpha(k-1)+2}} & \cdots & s_{x_1,y_{\alpha k}} \\ s_{x_2,y_{\alpha(k-1)+1}} & s_{x_2,y_{\alpha(k-1)+2}} & \cdots & s_{x_2,y_{\alpha k}} \\ \vdots & \vdots & \ddots & \vdots \\ s_{x_m,y_{\alpha(k-1)+1}} & s_{x_m,y_{\alpha(k-1)+2}} & \cdots & s_{x_m,y_{\alpha k}} \end{bmatrix} \tag{3.2}$$

and, likewise, for $\left(\boldsymbol{S}^{\top}_{n\times\beta}\right)^{(l)}$.

In order to sample each section, from its first row, a lowest cost path finding search is run to traverse the section to the last row, respecting a cost function (*e.g.* intensity). This is equivalent to finding a matrix $\boldsymbol{W}_{m\times\alpha} = \{w_{i,j}\} \in \{0,1\}$ for each $\boldsymbol{S}^{(k)}_{m\times\alpha}$ (and likewise for each $\left(\boldsymbol{S}^{\top}_{n\times\beta}\right)^{(l)}$) in the following way:

$$\arg\min_{\boldsymbol{W}} \sum_{i=1}^{m}\sum_{j=1}^{\alpha} w_{i,j}s_{x_i,y_j} \tag{3.3}$$

subject to:

$$\begin{aligned} &\sum_{j=1}^{\alpha} w_{i,j} = 1 \quad \text{and} \\ &|\sum_{j=1}^{\alpha} jw_{t,j} - \sum_{j=1}^{\alpha} jw_{t-1,j}| \leq 1, \quad t = 2,\cdots,m \end{aligned} \tag{3.4}$$

The problem, at each point (x, y), can be seen as finding the next best position until the search reaches the last row of the section. This problem is very similar to the *moving on a checkerboard* problem presented in [CLR$^+$01], which is well suited for discrete optimization with dynamic programming. This is illustrated in Algorithm DP, where ϑ is the sampling factor of the section.

DP$(\boldsymbol{S}, \vartheta)$

$\alpha = \vartheta m, \beta = \vartheta n$

// Vertical sampling

$\forall \boldsymbol{S}_{m \times \alpha}^{(k)} \in \boldsymbol{S}$ // $k = 1, 2, \ldots, \frac{n}{\alpha}$

// Solve Equation 3.3 for $\boldsymbol{S}_{m \times \alpha}^{(k)}$ to obtain $\boldsymbol{W}$

$\forall \boldsymbol{s}^\top \in \boldsymbol{S}_{m \times \alpha}^{(k)}$ // rows of $\boldsymbol{S}_{m \times \alpha}^{(k)}$

$\forall \boldsymbol{w}^\top \in \boldsymbol{W}$ // rows of $\boldsymbol{W}$

$\boldsymbol{f}_k = [\boldsymbol{s}_1^\top \boldsymbol{w_1} \quad \boldsymbol{s}_2^\top \boldsymbol{w_2} \quad \ldots \quad \boldsymbol{s}_m^\top \boldsymbol{w_m}]^\top$

// Horizontal sampling

$\forall \left(\boldsymbol{S}_{n \times \beta}^\top\right)^{(l)} \in \boldsymbol{S}$ // $l = 1, 2, \ldots, \frac{m}{\beta}$

// Solve Equation 3.3 for $\left(\boldsymbol{S}_{n \times \beta}^\top\right)^{(l)}$ to obtain $\boldsymbol{W}$

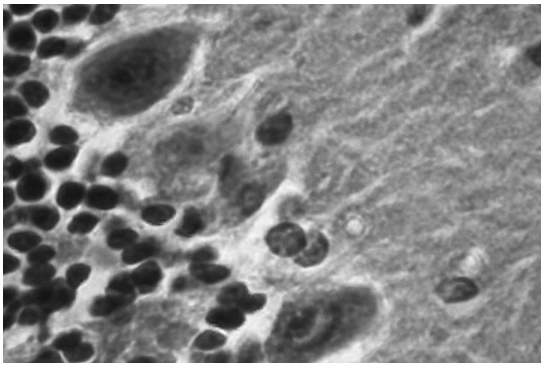
$\forall \boldsymbol{s}^\top \in \left(\boldsymbol{S}_{n \times \beta}^\top\right)^{(l)}$ // rows of $\left(\boldsymbol{S}_{n \times \beta}^\top\right)^{(l)}$

$\forall \boldsymbol{w}^\top \in \boldsymbol{W}$ // rows of $\boldsymbol{W}$

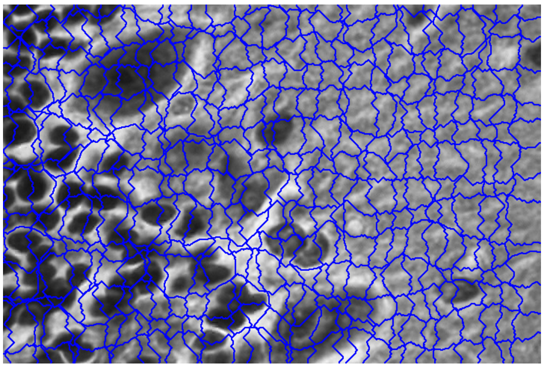
$\boldsymbol{f}_l = [\boldsymbol{s}_1^\top \boldsymbol{w_1} \quad \boldsymbol{s}_2^\top \boldsymbol{w_2} \quad \ldots \quad \boldsymbol{s}_n^\top \boldsymbol{w_n}]^\top$

$\boldsymbol{f} = \boldsymbol{f}_k \cup \boldsymbol{f}_l$ // $\forall k, l$

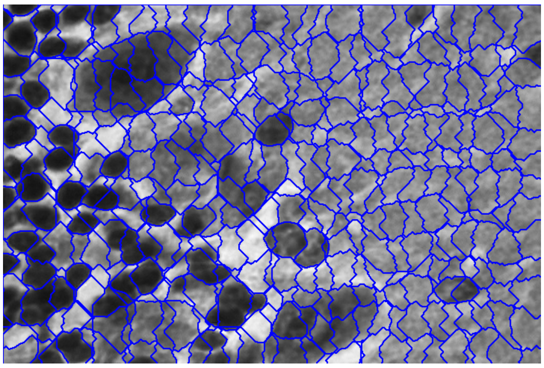
return $\boldsymbol{f}$



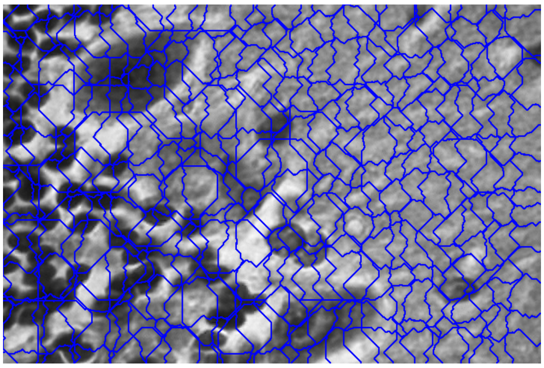
(a) A sample image.



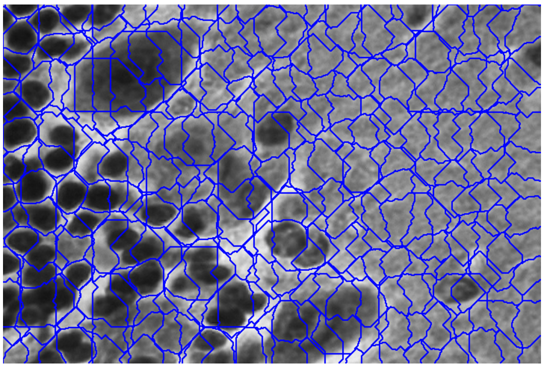
(b) Avoiding the edges.



(c) Attracting to the edges.



(d) Attracting to dark pixels.



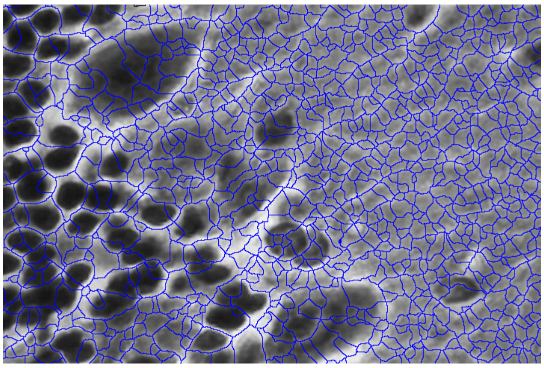
(e) Attracting to bright pixels.

Figure 3.1: Applying the dynamic programming approach to detect different features on a sample image.

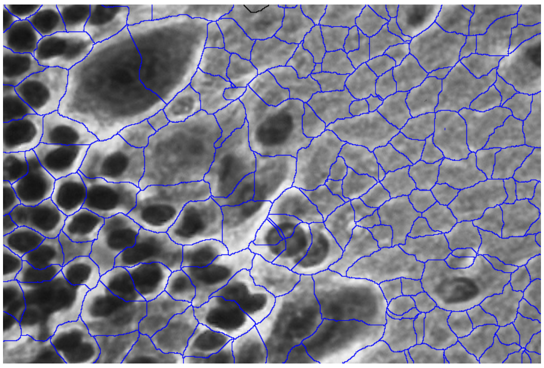
Figure 3.1 shows the results of applying this approach using different cost functions. The paths detected in Figure 3.1e seem to be good candidates to be used in the estimation of the bias field for this sample image.

3.2 Sampling Using the Watershed Transform

The Watershed Transform (WST) is a powerful mathematical morphology approach widely used in image segmentation. The watershed lines are built around figurative catchment basins where an object is supposed to be. A nice property of the watershed is that its lines reside on the background of the image (light background and dark objects). Due to noise this may not always be true, therefore, previously low-pass filtering the image may be required in order to prevent lines from crossing the objects. An example of applying the watershed by immersion approach [VS91] to the image in Figure 3.1a is shown in Figure 3.2.



(a)



(b)

Figure 3.2: Effect of smoothing on the results of the WST. (a) Watershed prior to smoothing. (b) Watershed after smoothing.

The use of the watershed to sample the image background has been successfully applied in [dAME$^+$11]. Unlike the sectioning approach, where the number of search sections has to be defined, watershed lines increase and decrease according to the smoothness of the input image.

3.3 Algorithm

This section describes the proposed algorithm to estimate the bias field. First, the bivariate polynomial used to estimate the illumination field is described, then the algorithm is presented and finally, the regularization and the stopping approaches are presented.

3.3.1 Estimating the Illumination Field

Given a vector of parameters $\boldsymbol{\gamma}$ and a polynomial model of degree d, the bias pixel at location (x, y) can be calculated as:

$$\boldsymbol{B}(x, y; \boldsymbol{\gamma}, d) = \sum_{i=0}^{i=d} \sum_{j=0}^{j=i} \gamma_{i,j} x^{i-j} y^j \tag{3.5}$$

which in matrix form can be rewritten as:

$$\boldsymbol{B}^{(\text{vectorized})} = \boldsymbol{V}\boldsymbol{\gamma} = \begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1 y_1 & y_1^2 & \cdots & y_1^d \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \\ 1 & x_h & y_h & x_h^2 & x_h y_h & y_h^2 & \cdots & y_h^d \end{bmatrix} \times \begin{bmatrix} \gamma_{0,0} \\ \gamma_{1,0} \\ \gamma_{1,1} \\ \vdots \\ \gamma_{d,d} \end{bmatrix} \tag{3.6}$$

where $h = mn$, and $\boldsymbol{V}$ is known as the Vandermonde matrix. For any polynomial of degree d, the number of columns of $\boldsymbol{V}$ is $r = \frac{(d+1)(d+2)}{2}$. In order to obtain $\boldsymbol{B}$, $\boldsymbol{\gamma}$ has to be found beforehand. The least squares fitting is used to find $\boldsymbol{\gamma}$ from the sampled image. The general fitting procedure is outlined as follows:

1. Let $\boldsymbol{f}$ be a vector of samples detected by one of the previously described approaches (*e.g.* DP).

2. Find $\boldsymbol{\gamma}$ that minimizes the following least squares problem:

$$arg \min_{\gamma} \frac{1}{2} \|\boldsymbol{V}\boldsymbol{\gamma} - \boldsymbol{f}\|_2^2 \tag{3.7}$$

In this work, QR decomposition with column pivoting (QRDCP) [GVL96, Hea02] is used to solve Equation 3.7. Since the condition number of $\boldsymbol{V}$, $\kappa(\boldsymbol{V})$, increases as the degree d increases, for high order polynomials, the solution might not be stable when classical approaches, such as the normal equations, are applied.

3.3.2 Correcting the Bias

The bias field is iteratively corrected, whereas the corrected image obtained at order $d-1$ is used as input at order d, and so on. The proposed bias correction approach is described in Algorithm CorrectBias.

CorrectBias($\boldsymbol{S}, \vartheta, d_{\max}$)

1 $\boldsymbol{S}_{\log} = \log(\boldsymbol{S})$

2 **for** $d = 1$ **to** $d_{\max}$

3 $\quad\quad \boldsymbol{f} = \mathrm{DP}(\boldsymbol{S}_{\log}, \vartheta)$ // or $\boldsymbol{f} = \mathrm{WST}(\boldsymbol{S}_{\log}, \vartheta)$ [VS91]

4 $\quad\quad \boldsymbol{\gamma} = \arg\min_{\boldsymbol{\gamma}} \|\boldsymbol{V}\boldsymbol{\gamma} - \boldsymbol{f}\|_2^2$

5 $\quad\quad \boldsymbol{B}_d = \boldsymbol{V}\boldsymbol{\gamma}$

6 $\quad\quad \boldsymbol{S}_{\log} = \boldsymbol{S}_{\log} - \boldsymbol{B}_d$

7 **return** Scale($e^{\boldsymbol{S}_{\log}}$)

In Algorithm CorrectBias, for DP, ϑ is the sampling ratio, and for the WST it is the smoothing factor. The maximum allowed polynomial degree is $d_{\max}$. In line

5, since the right-hand side (RHS) is a vector, $\boldsymbol{B}_d$ is mapped into the same size as the matrix $\boldsymbol{S}$. The SCALE function, at the last line, normalizes the image into the range $[0, 1]$.

3.3.3 Regularization

When fitting high order bivariate polynomials to an unevenly illuminated image, regularization must be used so that the resulting surface is as smooth as possible and faithful to the input image. The common approach for polynomial fitting regularization is the Tikhonov's method. The Tikhonov regularized version of Equation 3.7 is:

$$\arg\min_{\gamma} \left\| \begin{bmatrix} \boldsymbol{V} \\ \sqrt{\lambda}\boldsymbol{I} \end{bmatrix} \gamma - \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \right\|_2^2 \tag{3.8}$$

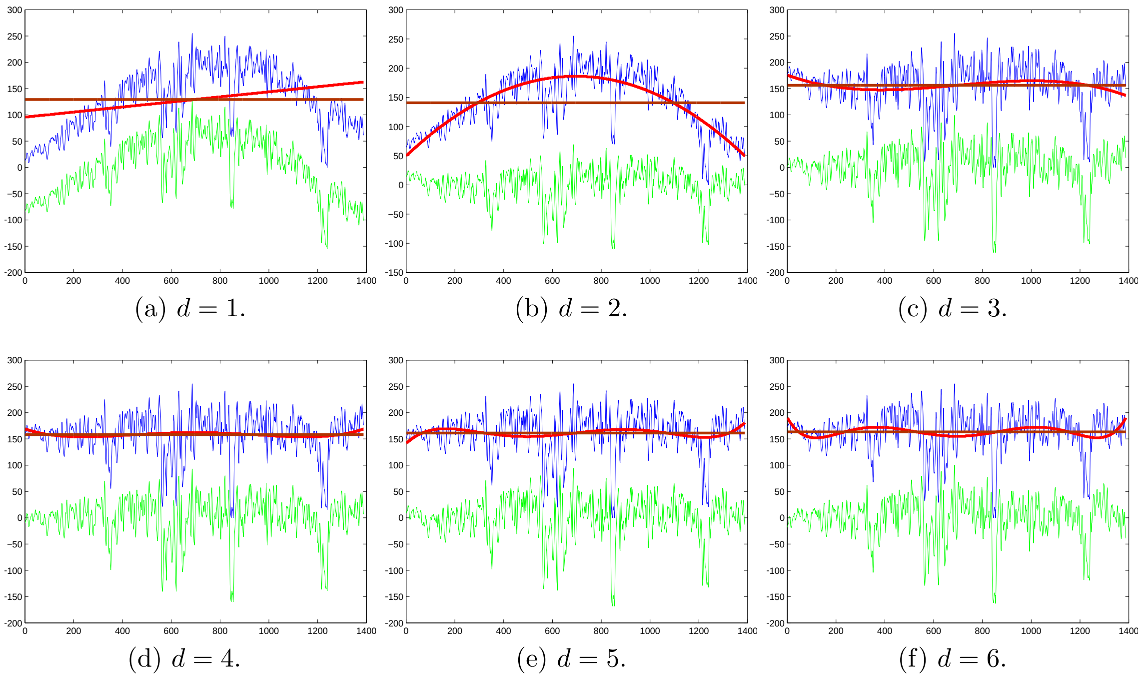
where λ is a balancing parameter between the fidelity term $\|\boldsymbol{V}\boldsymbol{\gamma} - \boldsymbol{f}\|_2^2$ and the smoothness of $\boldsymbol{V}\boldsymbol{\gamma}$, and $\boldsymbol{I}_{r\times r}$ is the identity matrix. Although Equation 3.8 is still differentiable (the closed form solution can be obtained), the second norm regularization discourages the sparsity of $\boldsymbol{\gamma}$. In this problem, sparse solutions are preferred in order to obtain smooth illumination fields. Moreover, it is very difficult to set λ to a fixed value for a single image and, the variation in the bias field complexity from one image to another, makes it even harder. The first norm has shown to have better regularization properties in image restoration [LBU12] and can be used as an alternative [CWB08]. However, formulating the problem by using the first norm as "regularizer" and the second norm as similarity measure makes it non-differentiable in some cases, requiring that sophisticated solvers be incorporated [MMOC11].

The smoothness of an estimated surface is inversely proportional to the polynomial degree in use, and the required regularization depends on the polynomial degree and the characteristics of the bias field. In the proposed approach, the sampling along with the gradual increase in the polynomial order in Algorithm CORRECT-BIAS, serves as an implicit regularization and ensures that the estimation of the bias is a smooth field. However, in the proposed approach, increasing the polynomial order continuously, may overfit the estimated bias field. Therefore, a stopping criterion is the key success in this approach in order to terminate the algorithm before converging to an over-fitted solution.

3.3.4 Stopping Criterion

For the sake of simplicity, let us assume there is a 1-D signal and 1-D polynomials. In addition, let the sampling process be ignored for a moment as it will, in any case, improve the fitting task by selecting proper samples from the image. Consider a 1-D signal without a bias. A good estimation for the "bias" of such signal is the signal's background, *i.e.* a "constant", and an approximation of the background of the signal can be the mean of the signal. In the proposed iterative approach, in order to stop at a given iteration, the similarity between the estimation and the mean of the signal is evaluated.

In Figure 3.3, a sample signal (blue curve) with unknown bias field is shown in the top-left sub-figure. The mean of the signal is shown in brown, the estimated curve is shown in red, and the corrected signal is shown in green. The polynomial degree, $d = 1, \cdots, 6$, increases from left to right and top to bottom. Note that, the corrected signal in order d is used as input in order $d+1$. When fitting is improving, the correction process heads toward a constant background, *i.e.* a horizontal straight line (see Figure 3.3d), because it is unlikely to "jump" from under-fitting to over-fitting without first meeting a good fit for the signal. This can be seen as a curve evolving into a straight line (perfect fit to the signal mean) and, when it starts over-fitting, it evolves from a straight line into a curve (Figures 3.3e and 3.3f). The



(a) $d = 1$. (b) $d = 2$. (c) $d = 3$.

(d) $d = 4$. (e) $d = 5$. (f) $d = 6$.

Figure 3.3: Iterative fitting by ever-increasing polynomial for 1D signal. Blue curve is the input signal, red is the estimated bias, brown is the mean of the signal, and green is the (unnormalized) signal after correction.

objective is to stop immediately before it starts to over-fit, in this example, at $d = 4$. This can be achieved, for instance, by comparing the sum of squared residuals (SSR), between the mean of the signal and the estimated curve, to a threshold. Nevertheless, because SSR has an unknown upper bound, it becomes impractical to set a threshold.

A well known measure of goodness of fit based on the SSR is the Coefficient of Determination, also known as R^2 (R-squared). Let μ be the mean of a 1-D discrete non-constant signal $\boldsymbol{f}$, and $\hat{\boldsymbol{f}}$ the estimated curve, then R^2 is defined as:

$$R^2 = 1 - \frac{(\boldsymbol{f} - \hat{\boldsymbol{f}})^\top (\boldsymbol{f} - \hat{\boldsymbol{f}})}{(\boldsymbol{f} - \mu)^\top (\boldsymbol{f} - \mu)} \tag{3.9}$$

where the numerator is the SSR, and the denominator is the total sum of squares (TSS).

A perfect fit between the estimated curve and the mean will happen when the explained sum of squares (ESS) is zero:

$$(\hat{\boldsymbol{f}} - \mu)^\top (\hat{\boldsymbol{f}} - \mu) = 0 \tag{3.10}$$

In practice, due to the iterative nature of the method, obtaining a perfect fit is not feasible. For that reason, a small threshold τ can be defined, and Algorithm CorrectBias will stop iterating when $R^2 \leq \tau$ instead of stopping at a static $d_{\max}$. In this work, τ is defined as 0.015. In practice, it is efficient to replace $\hat{\boldsymbol{f}}$ in Equation 3.9 by a low-pass filtered version of $\boldsymbol{f}$.

4 Experimental Results

4.1 Materials

In order to evaluate the proposed approach, a ground truth (GT) of 300 different images from three microscopy datasets was used. 113 images were selected from a malignant lymphoma dataset, and 150 images were selected from a mice liver dataset [AMHP09]. Each image is a 32 bit (color) TIFF of size 1388×1040. The remaining 37 images were selected from a dataset of various histology color images, 32 bit JPEGs of size 740×480 [CRCG11].

In order to generate a test set, the input images were converted into grayscale and, then, degraded by synthetic bias fields. Two methods were used to generate the bias fields: Gaussian beam and polynomial surface. In the former, Gaussian functions were centered at different positions in the input image. The spread in the x and y

(a) Gaussian biases.

(b) Polynomial biases.

Figure 3.4: Different biases created by 2-D Gaussian functions (top) and by 2-D polynomials of increasing order (bottom).

directions is the same (Figure 3.4a). In the latter, predefined parameters were used to generate different orders of polynomials (Figure 3.4b).

4.2 Evaluation

The proposed approach was compared to a set of different methods. These are: iterative least squares fitting based on the gradient (ITG) [ZGAG09, ZLK+09], improved illumination correction (IIC) [HM14], and surface fitting (SF) with polynomials of degree 5 using QRDCP. DP and WST sampling approaches were both included in the tests. The sampling criterion used in the DP was to attract to bright pixels (Figure 3.1e).

The parameters used were as follows: in Algorithm CorrectBias, ϑ was set to 0.05 for the DP approach, and 3 pixels as smoothing kernel radius for the WST approach. The maximum polynomial degree $d_{\max}$ was set to 15. In the ITG approach, the polynomial degree was set to 5, the exponent value of gradient distribution was set to 0.1, and iterated once (ITG produced better results when set to a single iteration).

The intuition was to understand how methods perform regarding objects' size and illumination field complexity. In Figure 3.5, the RMSE between the ground truth and the results of the evaluated methods are depicted. The results are summarized

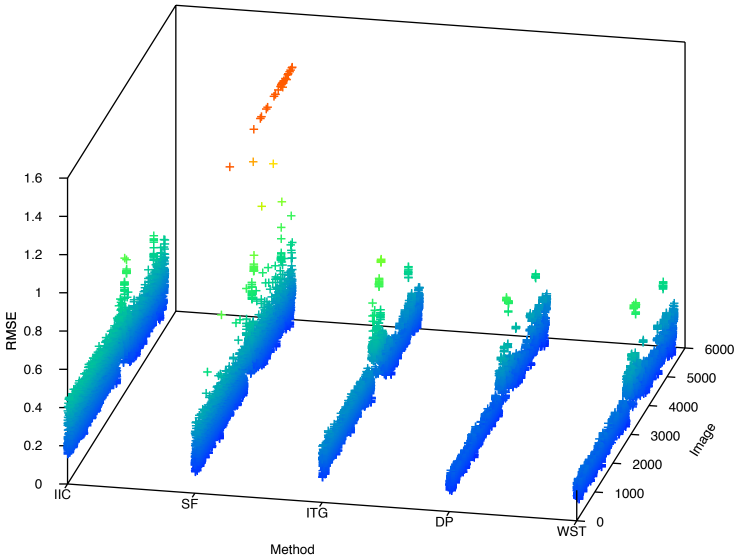


Figure 3.5: RMSE between the tested methods and the ground truth.

in Table 3.1. To cope with any normalization problem in the methods, the images were standardized before calculating the RMSE.

Table 3.1: Mean and Standard Deviation (STD) of the results presented in Figure 3.5 grouped according to object's size.

	IIC	**SF**	**ITG**	**DP**	**WST**
			SMALL		
Mean	0.2198	0.2250	0.2003	0.1930	**0.1896**
STD	0.0745	0.1545	**0.0541**	0.0665	0.0637
			MEDIUM		
Mean	0.2302	0.1829	0.2150	**0.1484**	0.1617
STD	0.0700	0.0630	0.0371	**0.0243**	0.0283
			LARGE		
Mean	0.3167	0.2994	0.3530	**0.2640**	0.2747
STD	**0.0994**	0.1177	0.1204	0.1099	0.1090

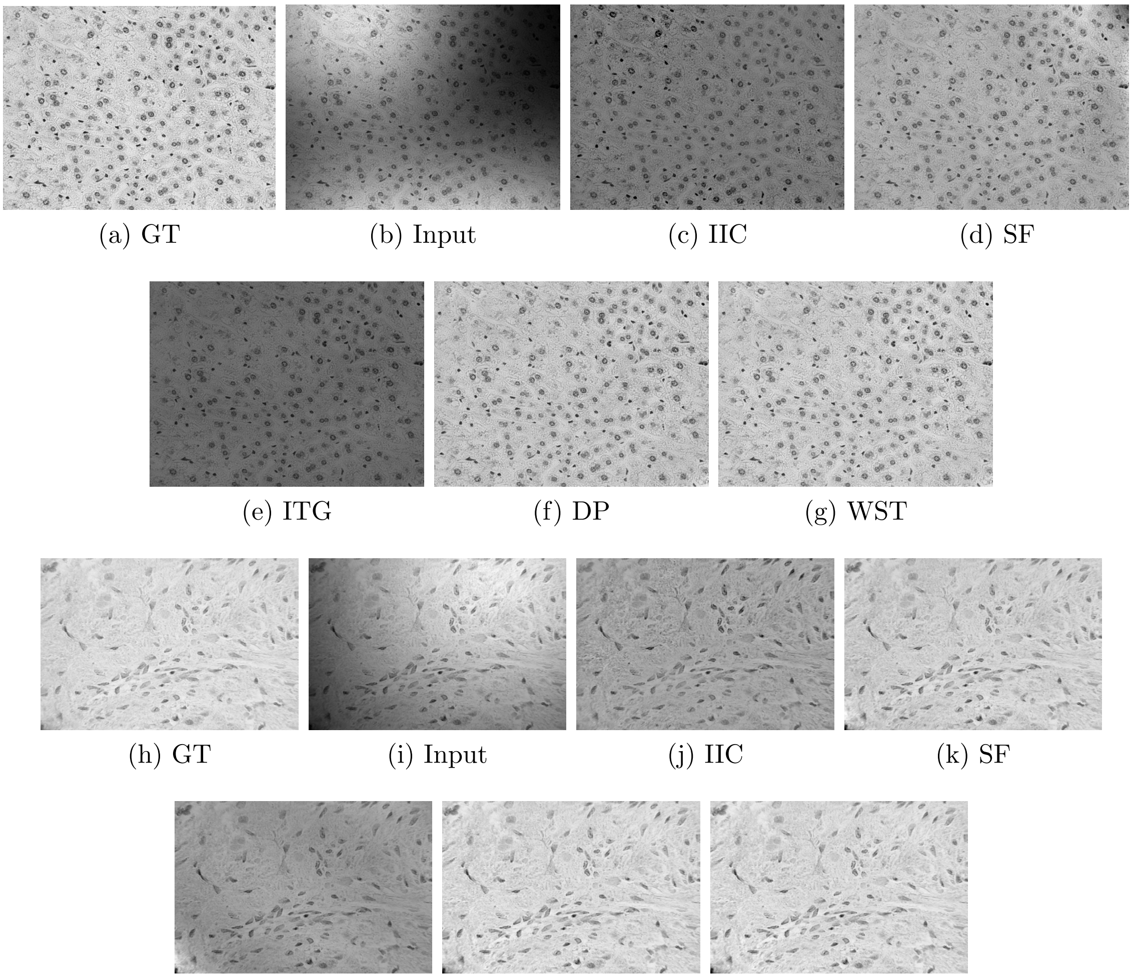
The time complexity of one iteration of the proposed algorithm can be calculated as follows: QRDCP requires $4mnu - 2r^2(m+n) + 4u^3$ flops [GVL96], where $u = \text{rank}(V)$; path finding by dynamic programming requires $3mn$ comparisons; subtraction and normalization requires $2mn$ flops. For WST, $5mn$ comparisons are required [RC07].

4.3 Discussion

The results in Figure 3.5 and Table 3.1 show the superiority of the proposed approach (DP and WST) when compared to the other methods. An important conclusion this figure highlights is that, the size of the object has an impact on the illumination correction because, when the background has more pixels than the foreground, less unwanted pixels are included in estimating the illumination. IIC shows a stable performance with very similar standard deviation in all cases. However, we experienced some images being washed out when a complex bias exists. DP and WST produced similar results. SF showed the highest variation among all, because fixing the polynomial degree, will produce unstable results under different bias fields and images. ITG showed competitive results only for small objects. As expected, in the presence of very large objects, at some point, all methods start to model objects in the estimated bias field but to different extents.

Examples of the results are shown in Figure 3.6. It can be observed that, the images recovered by the proposed approach are very uniform in terms of illumination. Note that the darkness in IIC and ITG is, in fact, not a problem if the corrected image is to be segmented. When calculating the RMSE in Table 3.1, this variation was neglected because the corrected images were standardized before comparison. Although the recovered image looks fine when using SF, it is not as good as the one obtained by the proposed approach. For example, in the image recovered by SF, there is a noticeable variation at the right corner of the image (Figure 3.6d).

The results of applying the proposed approach to three different images of microfilariae is shown in Figure 3.7. In this figure, it is clear that, the proposed method improves the illumination noticeably, especially in the first sample, where it is almost impossible to see the microfilariae at the bottom corners of the image.



(a) GT (b) Input (c) IIC (d) SF

(e) ITG (f) DP (g) WST

(h) GT (i) Input (j) IIC (k) SF

(l) ITG (m) DP (n) WST

Figure 3.6: Examples of the results of different methods.

As far as the parameters of the proposed method are concerned, for DP, ϑ is related to the size of the image, therefore, it can be estimated automatically. Setting it to a very high value (*e.g.* > 0.20), fewer pixels will be considered and vice-versa. It has been empirically found that setting ϑ to any value in the $[0.05, 0.12]$ produces good results.

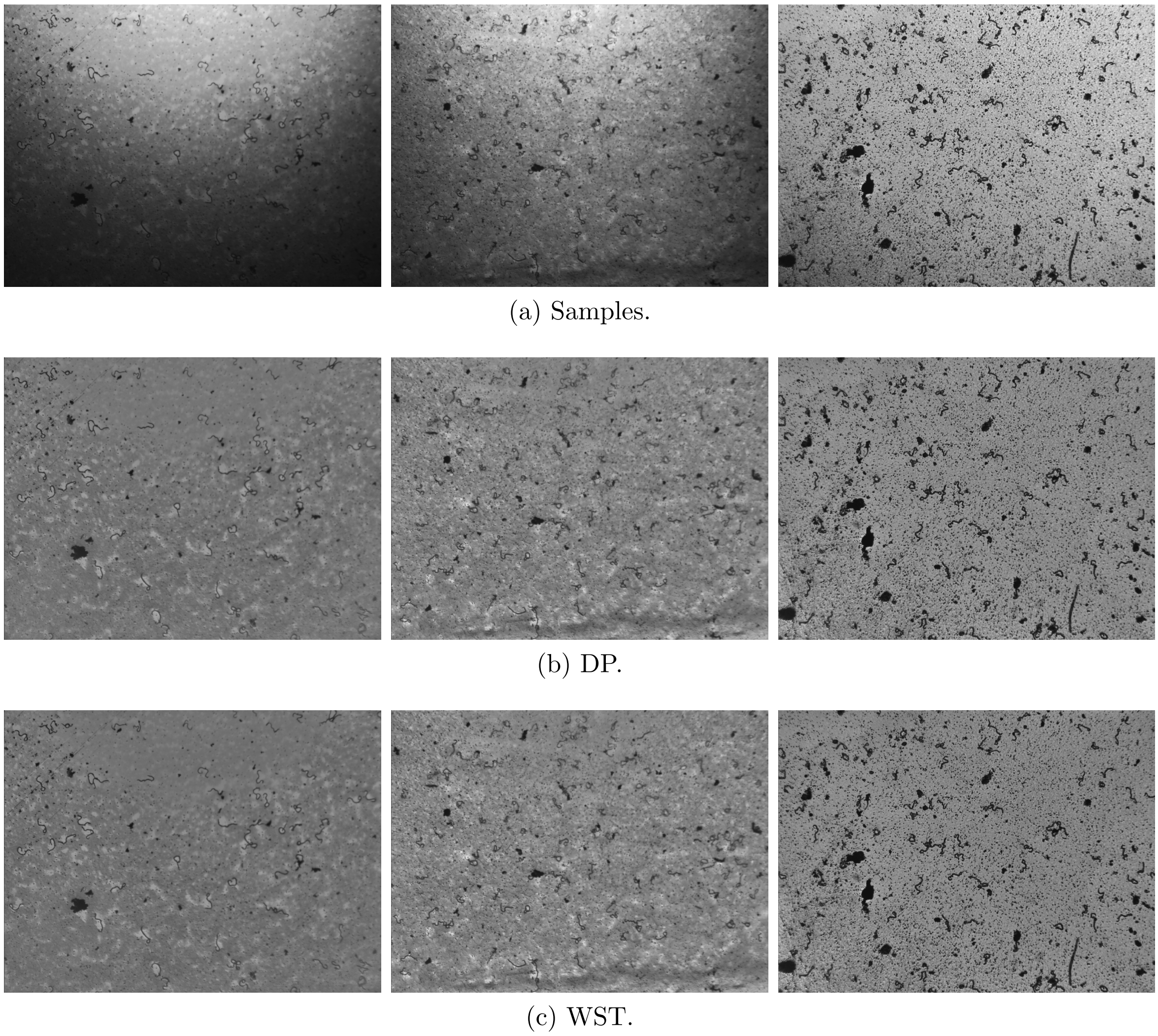


Figure 3.7: The results of applying the proposed approach on microfilariae images.

5 Conclusion

This chapter presented a new illumination correction approach. It samples the observed image and fits the samples to 2-D polynomials. Dynamic programming and the watershed transform were used as alternatives for sampling the input image. In the former, a vertical and horizontal sectioning approach is applied to the image, and the lowest cost path is calculated in each section. In the latter, the watershed is applied to a smoothed version of the input image. The least squares fitting with QRDCP is used to fit the sampled pixels to an increasing degree of 2-D polynomials.

The proposed approach was compared to a set of academic approaches with different correction mechanisms. Tests were performed using a collection of microscopy images of different objects' sizes and using different bias complexities. The results show that the proposed approach is very robust to complex biases and object size variations and produces the best results in all the tests.

4

OBJECT DETECTION

"No object is mysterious. The mystery is in your eye."

– Elizabeth Bowen, Anglo-Irish novelist

1 Introduction

An object in a 2-D image can been seen as a region (or union of regions) with pronounced boundaries and center [ADF10]. Even when contrast and other image intrinsic *non-idealities* are ignored, detecting objects is still a very challenging task. In addition to background clutter and partial visibility, objects bend, overlap, change size and appearance, and co-exist with many other similar or different objects (Figure 4.1).

(a) Mug with multiple objects. (b) Swan with background. (c) Microfilaria with background.

Figure 4.1: Different objects. The Mug and Swan images are taken from the ETHZ dataset [FTVG06]

Indeed, object detection is rich in literature but, up to our knowledge, there is no approach designed for microfilariae detection. In general, an object detection system, has four main steps (Figure 4.2):

- Feature extraction and analysis;
- Model building;
- Hypothesis generation;
- Verification.

Extracting features is the task of collecting features from objects, either by incorporating the domain of knowledge, or by using general purpose feature extractors. Thereafter, features are packed in a vector, known as *feature vector*, which is the input for model building and hypothesis verification tasks.

Features may need to be preprocessed and arranged in such a way that, they can be treated by the "machinery" that generates the model and the decision approach that verifies the hypotheses. This is important to avoid common problems like: curse of dimensionality, redundancy, inconsistency, and so on. In some cases, when several different features can not be aggregated into a single vector, due to numerical

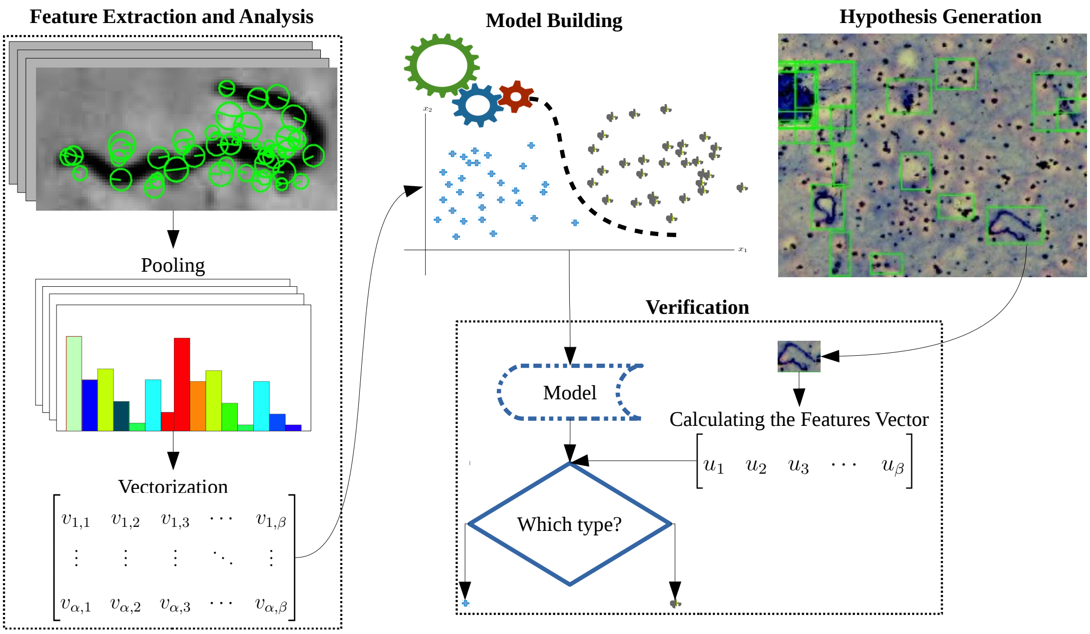


Figure 4.2: Object detection.

problems, they can be fed to independent model generators and then combined to form a single decision. Since features play a major role in designing a detection system [ZVRF12], they are discussed in more detail than the other tasks. Several feature extraction approaches are described in Section 2.

Model building is the process of using a machine learning approach to train a model from a set of labeled input feature vectors. In other words, building a model, is a synonym of finding the set of parameters of a function that better represents the input vectors. For instance, a linear regressor seeks the weights that minimize the residuals between the estimated and actual data, in a process called fitting whereas, a perceptron, searches for the weights that place a linear decision boundary, such that the generalization error is minimized, *i.e.* the probability of the perceptron misclassifying a sample is minimized. Among many different approaches, Support Vector Machines (SVM) are reviewed in depth in Section 3, as they are the core of many modern detectors.

Hypothesis generation, also known as object proposal generation, is based on finding bounding boxes where an object might exist. Apart from the model generation task, which can be the computational bottleneck in the detection system, identifying possible locations can be a very exhaustive task. Even though model generation may involve a training step, which can be very demanding, it is an off-line task, whereas searching for potential objects may require instant response from the system. Searching for objects, with different scales and orientations, requires not only scanning image pixels one by one, but also varying the bounding box size and aspect ratio. This is discussed in Section 4.

Once several possible locations, sub-images to be more precise, are generated, features are calculated in order to decide whether a sub-image contains an object or not, and which class it is from. In most of the cases, the algorithms for model generation, are provided with a verification system. Each sub-image is scored and, according to some criteria, a conclusion is drawn toward classifying the object. This task is briefly discussed in Section 5.

The types of features being extracted may limit the options available to build a suitable model. The nature of images and, also, the possibility of object size variation, have a clear impact on choosing the right hypothesis generator. In fact, extracted features also affect the hypotheses generation, and verification is dependent on the model that is being built, since the model created from the training data will be the key to the final discrimination process.

In the last decade, two main themes have dominated the object detection and recognition approaches [APTB$^+$14]. The first one treats objects in images as sub-windows [VJ04] which are scanned, encoded to feature vectors, and classified. The other one treats objects by their parts (regions, contours, and so on), such that, collecting information from each part produces a faster strategy than mere brute-force image sub-windowing [UvdSGS13, GLAM09]. This moves the bottleneck from the localization to the encoding and classification [UvdSGS13].

2 Feature Extraction

A feature is the elementary building block for any machine learning application. In computer vision applications, features come from the objects' appearance, geometry, and so on. For instance, the image gradient (Section 5.2.1 of Chapter 2) is an indicator of object structure, while color is an appearance measure. Features can be local or global, but have to be descriptive for the image content under some transformations. This section presents a brief description of the most common approaches. Comprehensive reviews can be found in [TM08, SMB00, LA08]. Other approaches will be briefly described in Sections 4 and 5, within the context of hypothesis generation and verification.

Local features are, in the simplest case, patterns that describe important geometrical, appearance, and/or morphological properties, like corners and regional maxima/minima. A general paradigm of local features detection involves the following steps [TM08]:

1. Detect keypoints (can be regions – *e.g.* blobs) using approaches like Harris and Hessian detectors, Laplacian of Gaussian (LoG - Section 5.2.1.1 of Chapter 2) [Lin98] (and its approximations [Low04, BTVG06]) and Maximally Stable Extremal Regions (MSER). Keypoints are expected to be robust, repetitive, distinctive and transformation invariant (mostly against translation, rotation, and scaling);

2. Define a patch around each keypoint in an invariant way. Usually, this is done by defining an elliptical region around the detected keypoint and, then, normalizing it to a circular canonical region;

3. Sample and normalize the content;

4. Compute a descriptor to encode the content in an easily comparable way. The role of the descriptor is to *fingerprint* the patch. This is important in order to yield unique features for each keypoint;

5. Matching of descriptors.

It is worth noting that, some recent approaches based on convolutional neural networks (CNN) [KSH12] have shown great improvements in object detection without explicitly following a feature detection approach. Recent CNN approaches produce the best results in many datasets [JSD+14, GDDM14]. The model generated by CaffeNet [JSD+14] from the ImageNet dataset [DDS+09] was tested on some microfilariae samples, but the results were not encouraging, and a specialized model from our own dataset was required. Unfortunately, it was not possible to carry out any further research in this trend due to hardware limitations in our lab.

Invariance to translation, rotation, and scaling are very important traits of a good feature descriptor. Affine-invariance is also required to cope with view change (see [MTS+05] for more details). Translation invariance is easy to achieve, since the whole domain of the image is used in the feature detection task. Rotation can be done by defining a canonical angle [Low99] and rotating the patch accordingly or, avoiding such reference angle as in [LSP05, WFW11, FWH12]. Since rotation invariance is method-dependent, it will be described within the methods. Scale invariance can be achieved by means of the scale-space theory, which is briefly described in Section 2.1. Global features are extracted from the whole image, and these are discussed in Section 2.3, whereas local features, which focus on detecting *salient* features, are presented in Section 2.4.

2.1 Scale-space

A scale-space of a function is a multi-scale representation which can be regarded as a level-of-details perspective [Lin94]. Looking at a tree from a far distance (coarse level) tells us what the tree looks like, physically, but it does not tell us anything about its leaf geometry, and the opposite will happen when looking at the tree from a very close distance (fine level). Between the coarse and fine levels, there is a continuous transition with different levels of detail.

A scale-space of a function $f : \mathbb{R}^d \to \mathbb{R}$ is defined by the mapping $\mathcal{J} : \mathbb{R}^d \times \mathbb{R}^+ \to \mathbb{R}$, such that:

$$\mathcal{J}(\boldsymbol{t}; \rho) = g(\boldsymbol{t}; \rho) * f, \tag{4.1}$$

where $\boldsymbol{t}$, of length d, is a vector of variables, ρ is a parameter that controls the scale, g is the scale kernel function, and $*$ is the convolution operator. It has been shown in [Lin94] that the only g that is acceptable under some admissibility conditions is the Gaussian kernel, and the only kernel to be used to perform scale-space derivative, is the Gaussian derivative (see Section 5.2.1.2 of Chapter 2).

2.2 Keypoints and Blobs Detectors

A point-of-interest, also known as keypoint, is a point in the 2-D image grid which is noticeably different from its neighbors (two-dimensional change [SMB00]). Exploiting the structure around a point is very helpful to investigate the *keyness* of a point. This is the central idea found in interest point detectors.

2.2.1 Harris and Hessian Detectors

Harris keypoint detector [HS88] is an extension of the idea presented by Moravec [Mor77], where the keyness of a point is approximated using the auto-correlation. Within a window being shifted to several directions, the Harris keypoint detector evaluates the shape of the distribution of the gradient around the center point of the window. By analyzing the principal components of this distribution, several keypoints can be categorized. For instance, corner points are known to produce large principal components. When shifting the window by u in the x direction and v in the y direction, the auto-correlation can be approximated as:

$$[u \ v] \underbrace{\left(\sum_{x,y} w_{x,y} \begin{bmatrix} L_x^2 & L_x L_y \\ L_x L_y & L_y^2 \end{bmatrix} \right)}_{\mathcal{M}} \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.2}$$

where $w_{x,y}$ is an element in the window $\boldsymbol{W}$. This window is generated by a Heaviside-like step function (*i.e.* has support only inside the window and zero otherwise) or a Gaussian. $\mathcal{M}$ is called the Harris matrix (moment matrix), which is a good descriptor for the local structure of $\boldsymbol{W}$ around a point, and that is why it is also called the *structure tensor* matrix. In many cases, $\boldsymbol{W}$ is chosen to be a Gaussian function to penalize the proximity of points to the center.

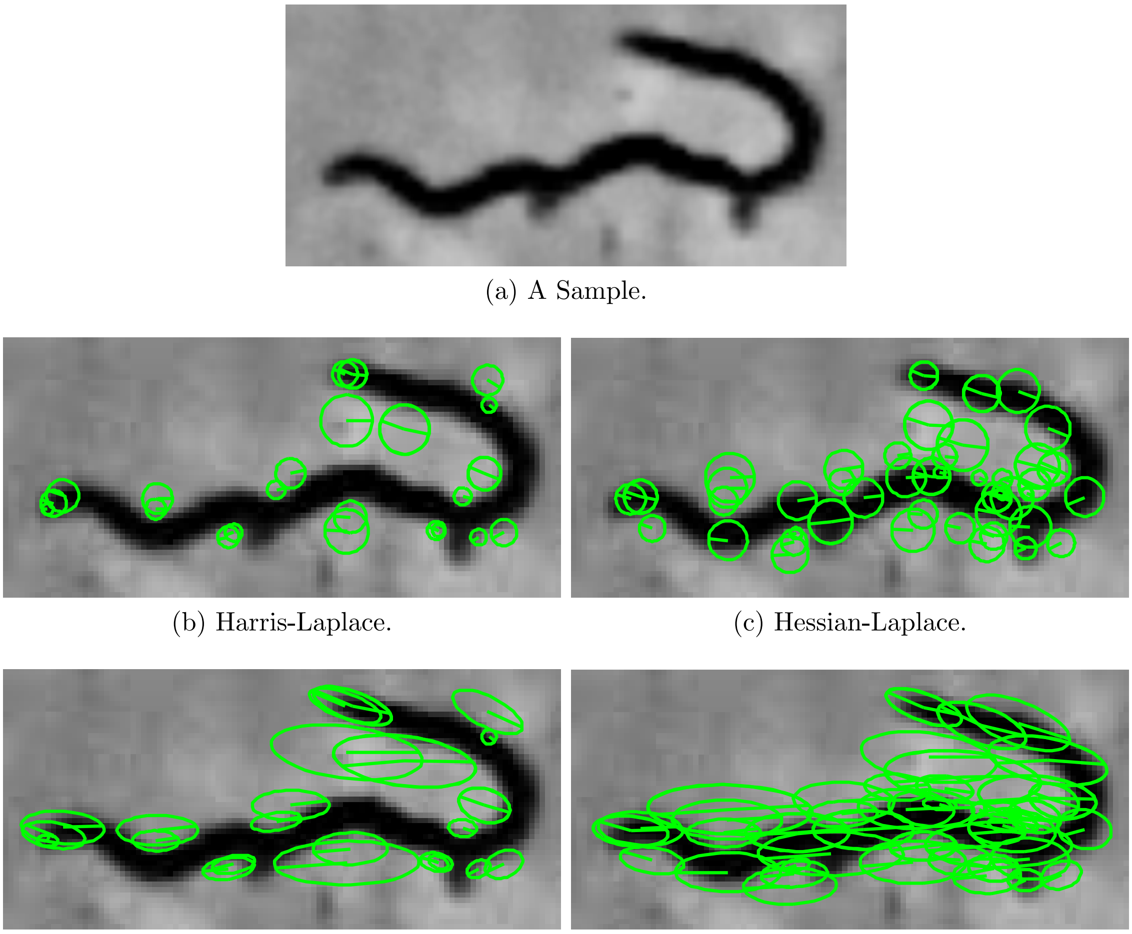
Eigenanalysis (and simply the rank analysis of $\mathcal{M}$ [MS04]) is a good way to distinguish regions around points. For example, as stated above, a corner will yield two high eigenvalues for both principal components of $\mathcal{M}$, and a flat region will show very small eigenvalues. Likewise, a line-edge shows a large value in one principal component only. Usually, the detection is followed by a non-maximum suppression step [Can86] to avoid redundant and weak points.

In order to incorporate the scale-space to obtain a scale-invariant Harris detector, the Harris-Laplace was developed [MS04]. First $\mathcal{M}$ is transformed to scale-space, *i.e.* multi-scale Harris, by replacing the derivatives in $\mathcal{M}$ by Gaussian derivatives with different scales (differential scales), and replacing $\boldsymbol{W}$ by a scale-varying Gaussian (integration scale). Increasing both the differential and integration scales while tracking the maximum LoG, yields an iterative way to identify the scale of the patch around the interest point. This is because the LoG works as a matching filter, *i.e.* wherever the scale fits the region, the LoG becomes maximum (Section 5.2.1.1 of Chapter 2). This is the main idea of an early work of blob detection by Lindeberg [Lin98].

Since the change in the scene view produces anisotropic scale variation, scale-invariant Harris produces different patches for the same region under different views. To cope with this limitation, Mikolajczyk and Schmid [MS02, MS04] transformed $\mathcal{M}$ to facilitate the affine deformations. Intuitively, the difference between the scale-invariant and the affine-invariant is similar to the difference between the Euclidean and the Mahalanobis distances, where, in the latter, the covariance matrix plays a major role to incorporate the anisotropy of the distance. The region detected by the affine-Harris is an ellipse collinear with the estimated view, which is usually mapped "back" to a circular patch to reverse the affine deformation.

Likewise, the Hessian matrix $\mathcal{H}$ (Equation 2.44 in Section 6 of Chapter 2) is used to model the structure around a point [Bea78]. The determinant can be used to detect keypoints in a similar way as in the Harris detector.

In Figure 4.3, a microfilaria sample is tested with the Harris-Laplace and Hessian-Laplace, and Harris-Affine and Hessian-Affine detectors. The orientation is estimated using the method presented in Section 2.4.1.2. Since microfilariae have no Harris features, the Harris responses were concentrated on the concave and convex borders of the microfilaria. This is where the cornerness is high. The detected regions are irrelevant, because they do not exploit the microfilaria structure. On the other hand, the Hessian detector is attracted to corners, and to some other relevant regions. In fact, further investigation of the Hessian showed that, it is a very helpful ground to design an effective thin-structure detection approach within the differential geometry framework, as discussed in Chapter 5.



(a) A Sample.

(b) Harris-Laplace.

(c) Hessian-Laplace.

(d) Harris-Affine.

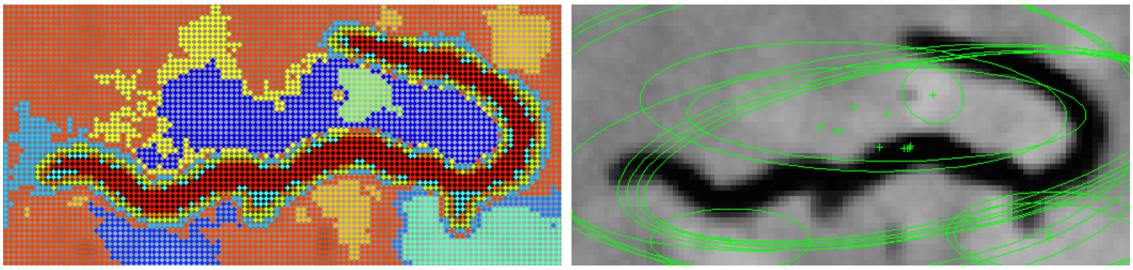
(e) Hessian-Affine.

Figure 4.3: Covariant regions detected by various detectors of the sample image. The orientation is estimated using gradient orientation as descried in Section 2.4.1.2.

2.2.2 Maximally Stable Extremal Regions

Maximally Stable Extremal Regions (MSER) is a region detector [MCUP04]. Intuitively, it uses the same principle of watershed by immersion [VS91] to obtain image regions from segmentation. That is also equivalent to describing the image using the level sets principle [KZBB11]. For an image $\boldsymbol{S}$ and an ordered intensity level $l \geq 0$, thresholding $\boldsymbol{S}$ by increasing l with a predefined increment factor k, will produce a series of binary images. These images can be seen as the time frames of the "evolution movie" of the regions. At low thresholds, small spots will emerge, which will evolve (grow) into regions as the threshold increases. At some thresholding levels, some regions will merge. When approaching high thresholds, the whole image will be a single region. Some connected components of these images do not change (stable) for a range of consequential thresholds, these are *extremal* regions. All pixels in an extremal region are noticeably different (darker or brighter) from all pixels of the border of the region. However, additional validations can be tested to ensure stability in the regions based on the maximum and minimum variation in the region. For a stable region to become *maximally stable*, it should have a local maxima (or minima for dark regions). In [MCUP04] it has been shown that MSERs are affine-covariant, but they are very sensitive to blurring [MTS+05]. For that reason, MSER is coupled with edge information, *i.e.* the gradient [CTS+11, KZBB11]. Since MSER is a region detector, regions are mapped to a common shape, typically ellipses as shown in Figure 4.4. These ellipses are then converted into circular patches, treatable by region descriptors like LIOP (Section 2.4.7) and SIFT (Section 2.4.1).

This method can be implemented in a similar way as the watershed [MCUP04], based on sorting the pixels according to intensity and, then, detecting the connected components using an efficient union finding algorithm. A fast implementation of



(a) MSERs' regions. (b) MSERs' ellipses.

Figure 4.4: The MSER of the sample image in Figure 4.3a.

this detector can be found in [NS08]. Color and 3-D extensions of this detector can be found in [For07] and [DB06], respectively.

2.3 Global Features

2.3.1 Histogram Pooling

In this approach, pixels intensities are pooled into a histogram (Section 3 of Chapter 2). The histogram is robust to object rotation and scale, but it is susceptible to illumination and contrast variations. It can also be applied to multiple channels [SB91] and produces reasonable results in many problems. It can be densely applied to the image by concatenating the calculated sub-histograms from sub-windows of the image. The spatial pyramids approach [LSP06] can be used to pool local histograms in a more robust descriptor. Despite of the simplicity of this approach, it has been shown that encoding multiple channels in the histogram is very helpful even for difficult problems like image segmentation [Mig08]. However, one has to be careful when dealing with several color spaces, since they respond differently to light shift and scale variations [vdSGS10], and may contain redundant information.

2.3.2 Co-occurrence Matrix

The co-occurrence matrix, also known as gray level co-occurrence matrix (GLCM), can be seen as an extension of the 1-D intensity histogram. In grayscale images, the histogram does not provide any spatial information. Additional information about the vicinity of pixels can be taken into account by using the co-occurrence matrix [HSD73]. This approach is widely used for texture analysis, but it is also used in image analysis in general [GRB08, LY08].

In this approach, the co-occurrence matrix contains information on how intensity values co-occur in the image. In other words, it accumulates the frequency of occurrence of two intensity values in a given neighborhood. Obviously, one needs to know the size of the neighborhood (k_x, k_y) and the number of the intensity levels.

Formally, given an image $\boldsymbol{S}$, the co-occurrence matrix $\boldsymbol{C}$ can be calculated as:

$$c_{l_1,l_2} = \sum_{x=1}^{m}\sum_{y=1}^{n} \begin{cases} 1 & \text{if } s_{x,y} = l_1 \text{ and } s_{x+k_x,y+k_y} = l_2 \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

for any two intensity levels l_1 and l_2, c_{l_1,l_2} is the element in $\boldsymbol{C}$ at position (l_1, l_2). $s_{x,y}$ is the pixel at location (x, y), and $k_\square$ is a predefined displacement constant in the direction $\square$. After calculating $\boldsymbol{C}$, it is normalized in a similar way as the histogram normalization in Section 3 of Chapter 2, by dividing $\boldsymbol{C}$ by the total number of co-occurrences, to obtain a discrete joint probability matrix.

Several co-occurrence matrices can be obtained for different orientations. These matrices are then combined, and statistical features like correlation, contrast, entropy, and others are used to form a feature vector. A color version concept is introduced in [HKM+97].

2.3.3 Histogram of Oriented Gradients

Mapping image gradient magnitude with respect to local gradient orientation in a histogram is called Histogram of Oriented Gradients (HOG). The original work is published in [DT05] for pedestrian detection. Since then, several improvements have been added, *e.g.* [ZYCA06], and it became the core of many state-of-the-art methods like the part-based HOG [FGMR10]. This deformable part-based HOG model solved some of the problems found in the original HOG. In fact, it is designed to cope with objects' deformation and, for that reason, and due to its great success (lifetime prize winner of Pascal challenge [EGW$^+$10]), it will be evaluated for our problem. In addition, Bharath *et* al. [HMR12] proposed a whitening HOG (WHOG) using Linear Discriminant Analysis (LDA). Both approaches are combined to produce a robust interactive real-time object detector presented in [GHR$^+$14].

In order to calculate the HOG, the gradient vector is calculated on the whole image. Thereafter, the gradient response is divided into overlapping cells (sub-windows). A histogram, based on the gradient orientation of each cell (cell histogram), is calculated. The bins of this histogram are the orientations of the gradient, and the magnitude of a bin is the accumulated gradient magnitude of a given orientation. In practice, orientations are quantized to a few number of bins (*e.g.* 9) and interpolation is used to split the gradient magnitude between overlapping bins.

To normalize against contrast variation, blocks are formed by cells (*e.g.* 2×2 cells), and histograms of cells in each block are concatenated to form a single histogram (block histogram), which is then normalized. An example of this feature detector response is shown in Figure 4.5. The response of HOG to the object has a tendency to follow a specific orientation, forming a pattern of local "gradient flow". Encoding these features into one single histogram produces a rich descriptor of the shape structure.

In our problem, two main obvious limitations of HOG are the rotation variation and the correlation between the feature vector size and the image size. A solution to the former problem can be by either running HOG in different sampled poses or by incorporating the polar transform as in [LSS+14] when calculating the gradients. A widely used solution to obtain a fixed size feature vector is scaling all training images to the same size. In practice, due to the high intra-class variation of the microfilariae aspect ratio, it was impractical to define a fixed size. This problem is solved in [FGMR10] by clustering the training set according to the aspect ratios of the images.

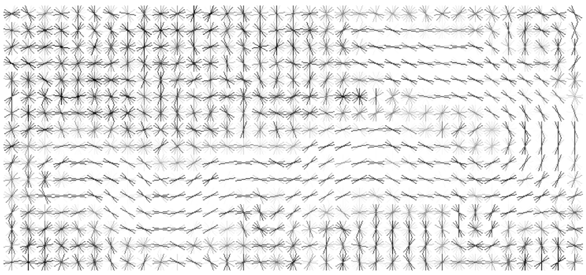


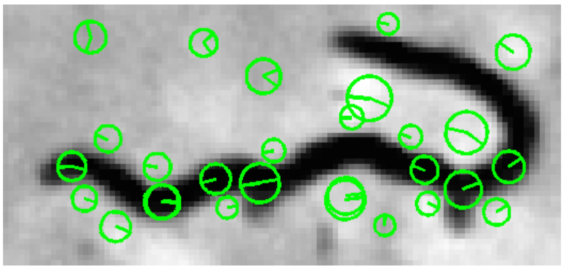
Figure 4.5: The HOG response of the sample image in Figure 4.3a, where the cell size is 3×3 pixels. The shape of the response determines the strength of the gradient magnitude with respect to the gradient orientation. Those stars with more spikes show no clear orientation.

2.4 Local Features

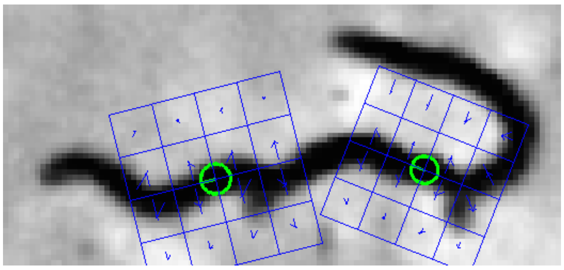
This section describes two different groups of descriptors. The first group is described in Subsections 2.4.1-2.4.4. It is a set of approaches that rely on the image gradient to describe the image content. The second one is a set of calculated descriptors based on the relative intensity of a pixel and its neighbors (*i.e.* equal, greater, and smaller), these are discussed in Subsections 2.4.5 - 2.4.7.

2.4.1 Scale Invariant Feature Transform

It is one of the most influential works in feature detection developed by David Lowe [Low99, Low04], and widely-known in the computer vision community by its abbreviation, SIFT. Its simplicity and efficiency in detecting interest points under scale variation makes it very popular. Nevertheless, SIFT descriptor is just a histogram of oriented gradients calculated around a keypoint (Figure 4.6), SIFT algorithm involves detecting keypoints and calculating feature vectors. Its steps are briefly described as follows.



(a) Interest points detected by SIFT.



(b) Descriptors of two selected points.

Figure 4.6: The SIFT keypoints and descriptors. The orientation of the keypoint is indicated by the pose of the descriptor frame. The blue arrows show the dominant direction of the gradient orientation inside the sub-windows.

2.4.1.1 Detecting Keypoints In order to detect scale-invariant keypoints, Lowe used the scale-space theory [Lin94] to generate a space of scaled images, where the LoG is approximated by the Difference of Gaussian (DoG). Therefore, detecting keypoints boils down to detecting maxima and minima in the DoG scale-space.

In order to calculate the locations of keypoints, a set of volumetric octaves is created (Figure 4.7). The image at the top of an octave is the original one, while the one at the bottom is the most blurred version. In-between them, there is a set of increased blur versions of the original image. The blurriness increases uniformly when traveling from the original to the most blurred version. This way, each octave is a 3-D function of (x, y, σ), where σ is the standard deviation of the Gaussian. Typically, each octave contains 5 images (levels of σ), and σ is doubled from one level to another. To create another octave, the image's size of the parent octave is halved. The recommended number of octaves is 4. These octaves are refereed to as the *Gaussian octaves* which are used to generate the *DoG octaves*.

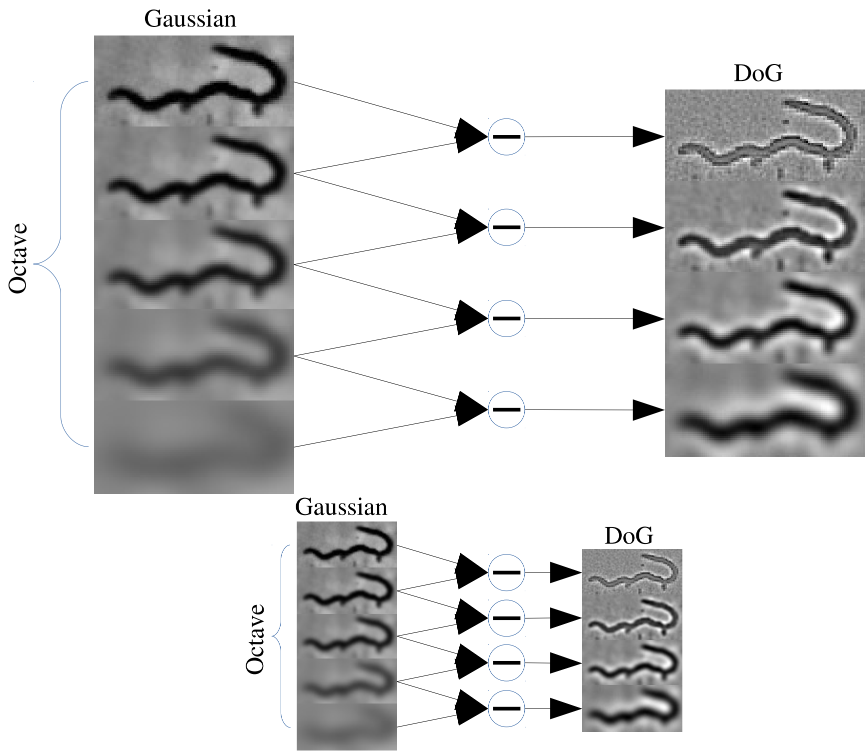


Figure 4.7: SIFT octaves. Gaussian Octaves are shown in the left, while the DoG ones are shown in the right. The initial σ was 1 and doubled from top to bottom inside the Gaussian octave. The image's size at the lower octave is half of the parent's at the top.

In order to create a DoG octave, the images of the corresponding Gaussian octave are subtracted as shown in Figure 4.7. In order to detect maxima and minima in the DoG, each pixel in the DoG octave is compared to its 26 neighbors (*i.e.* $3 \times 3 \times 3 - 1$) and, if it is greater or lower than all of them, then it is selected as a keypoint. In order to obtain subpixel resolution, the DoG octave is approximated by a second order

Taylor expansion [BL02]. Thresholds are used to filter out low-contrast keypoints and the removal of edge keypoints is achieved using curvature analysis by means of eigenanalysis of the Hessian matrix.

2.4.1.2 Orientation of Keypoints To obtain a rotation invariant descriptor around each point, it is important to define a reference angle to adjust the descriptor orientation. Within a neighborhood, a histogram of oriented gradients is calculated as in the HOG descriptor around each keypoint. A histogram of 36 bins, where each bin covers 10 angles, is created. Each bin accumulates the gradient magnitude weighed by a circular Gaussian window of 1.5 of the keypoint scale to penalize the proximity to the keypoint. The orientation of the keypoint will then be the peak of the histogram. Any peak that is at least 80% of the highest peak is assigned a new keypoint. Therefore, multiple keypoints may share the same position but with different orientations. In fact, in [FWH12], this step is said to decrease the effectiveness of SIFT and other SIFT-Like approaches such as Speeded Up Robust Features (SURF).

2.4.1.3 Calculating the Descriptors In order to calculate the descriptor, the gradient magnitude and orientation of a predefined sized window around the keypoint is calculated. The coordinates of the descriptor and the orientations of the gradient are adjusted with respect to the keypoint orientation to obtain rotation invariance (notice the orientation of the frame as well as of the radii in Figure 4.6). The descriptor window is divided into 4×4 sub-windows as shown in Figure 4.6b. A histogram of 8 bins is then used to pool the Gaussian weighed gradient magnitude inside each sub-window. The final feature vector length will be 128 resulting from concatenating the 4×4 sub-windows' 8-bins histograms. The vector is normalized to unit length to increase the robustness to illumination changes.

Different extensions of SIFT have been proposed in the literature, concerning reduction of the feature vector length [KS04] (PCA-SIFT), extending it to color spaces [AHF06, vdSGS10], applying it on grid [FFP05] (Dense-SIFT – Appendix B), and so on. In addition, Color Independent Components based SIFT Descriptor (CIC-SIFT) [AHRwC10] relies on constructing an adaptive color space using the Independent Component Analysis (ICA) transformation and then applying SIFT on each component. Images in the dataset are categorized into groups, then the ICA transformation matrix is learnt from all images in each group. Each group will have its transformation matrix. This matrix is then used to transform each RGB component in the input image, and SIFT is calculated on these transformed components.

According to [MS05] (and also to our preliminary evaluation) SIFT is superior to many different descriptors and, recent studies [UvdSGS13], report good performance of SIFT applied to the opponent color space (Opponent-SIFT)[vdSGS10], which will be included in the tests in Chapter 6.

2.4.2 Rotation Invariant Feature Transform

The Rotation Invariant Feature Transform (RIFT) is proposed in [LSP06] as a SIFT descriptor without explicit reference angle calculation. In RIFT, given a patch, the rotation invariance is achieved by calculating the HOG on concentric equidistant circular regions spanning the patch. The orientation of the gradient is normalized with respect to the center of the patch to achieve rotation invariance. Such normalization might reduce the distinctiveness of the features [FWH12].

2.4.3 Shape Context

Shape Context (SC) [BMP02] was first introduced to match shapes using sets of points gathered from their sampled edges or contours. SC is a histogram-based descriptor, in which each point has a 2-D histogram that describes its *relationship* to other points in the shape model. Formally, given a point set T with size n, $\forall t_i \in T$, SC calculates the Euclidean Distance and angles between t_i and t_j, $\forall t_j \in T$ and $i \neq j, \forall i, j = 1, \cdots, n$. The union of all points' histograms gives a rich description of the shape. Putting it simply, by binning the points in a log-polar space and calculating the distances and the angles of the points, an entry in the histogram h of a point t_i is then represented mathematically as a function $h_{t_i}(a, b)$, where $a \leq r$ and $b \leq \theta$. The log radius r is the height of the histogram and corresponds to the number of diameters in the log-polar diagram. The width of the histogram is θ and it corresponds to the number of circular sectors in the log-polar diagram. The width of each section is determined by the upper u and lower l limits of the log-polar space. This way SC has four parameters: r, θ, l and u. For the sake of simplicity, let $h_{t_i}(a, b)$ be represented as $h_{t_i}(k)$ $\forall k \leq r \times \theta$. Correspondent points in two similar shapes should have similar shape contexts. When matching two shapes, the use of χ^2 test statistic (Section 3.1.1 of Chapter 2) is adequate for these histograms.

2.4.4 Basic Image Features

In order to calculate the Basic Image Features (BIFs), Crosier and Lewis [CG10] proposed the decomposition of an image into "basic" features, such as flat area, ridge, and so on. For that purpose, the image is convolved by six Gaussian derivative kernels of orders 0 to 2, to produce six different images. These are $\boldsymbol{S} * \{\mathcal{G}, L_x, L_y, L_{xx}, L_{yy}, L_{xy}\}$.

Each pixel in the source image is classified as one of seven classes according to the information collected from these images as detailed in algorithm CalcBIFs:

CalcBIFs($\boldsymbol{S}, \sigma, \epsilon$)

1 $\mu_{\alpha\beta} = \sigma^{\alpha+\beta}(\boldsymbol{S} * L_{\alpha\beta})$ // scale normalized kernel

2 $\lambda = \mu_{20} + \mu_{02}$

3 $\gamma = \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}$

4 **return** $\max\left\{\epsilon\mu_{00}, 2\sqrt{\mu_{10}^2 + \mu_{01}^2}, \pm\lambda, 2^{\frac{1}{2}}(\gamma \pm \lambda), \gamma\right\}$

where each term in the last line is a BIF.

In line 1, L is calculated using Equations 2.37 - 2.41, and $\alpha, \beta = 0, 1, 2..$

In order to obtain a fixed size vector, BIFS is applied to different scales (*e.g.* 4) of σ, while ϵ controls the noise tolerance for flat regions. A histogram of $7^4 = 2401$ is then created and the occurrence of the features columns (BIFs-columns) are counted for each bin.

2.4.5 Local Binary Pattern

The Local Binary Pattern (LBP) is a feature descriptor designed for texture analysis [OPM02]. A circular mask of radius r is centered at each pixel of the image. The neighbors are then subtracted from the center pixel producing positive and negative responses, which are then encoded to binary (*i.e.* {0,1}). Encoding these values in a word, and calculating the occurrence of the words in a histogram generates a feature descriptor of the image. However, as image rotates, these words will change as well. Rotation invariance can be achieved in different ways [AMHP09, RLC$^+$10]. Perhaps the most intuitive and simple way is by bit-wise-circular shifting the binary sequence, and taking the minimum as unique descriptor [AMHP09]. The shifting direction depends on the numbering order of the neighbors, *i.e.* clock- or counterclockwise.

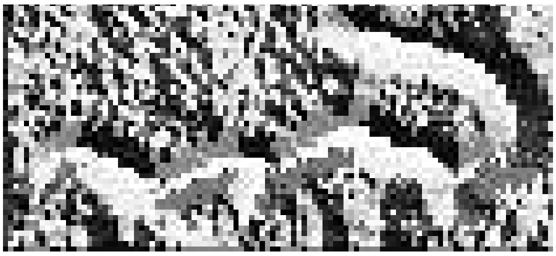


Figure 4.8: LBP calculated for the sample in Figure 4.3a.

For a pixel $s_{x,y}$, a circular neighborhood can be created by using the polar transform. The position of a neighborhood pixel s_{x_k,y_k} is calculated by:

$$\begin{aligned} x_k &= x + r\cos\left(\frac{2\pi k}{K}\right) \\ y_k &= y - r\sin\left(\frac{2\pi k}{K}\right) \end{aligned} \tag{4.4}$$

where $k = 1, \cdots, K$ is each of the K neighborhood pixels. When (x_k, y_k) does not fit in the image's discrete grid, interpolation is used. The binary feature vector for $s_{x,y}$ is obtained by decoding the calculated word into its binary sequence:

$$\sum_{k=1}^{K} [s_{x,y} \geq s_{x_k,y_k}] 2^{k-1}, \tag{4.5}$$

where $[\square]$ is the Iverson bracket. The response of LBP for the sample in Figure 4.3a is shown in Figure 4.8.

2.4.6 Binary Robust Independent Elementary Features

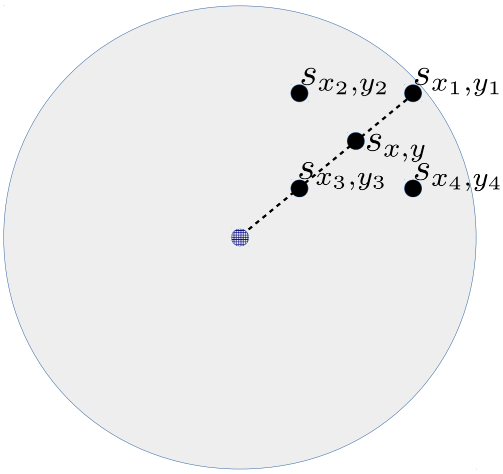
In a similar way to the LBP principle, Calonder *et* al. [CLSF10] developed the Binary Robust Independent Elementary Features (BREIF) approach. Like LBP, it describes a region around a keypoint by a binary vector. In order to do that, BREIF first samples k pairs of pixels, and then compares them. Similar to Equation 4.5, given a patch surrounding a keypoint, BREIF's word is calculated as:

$$\sum_{k=1}^{K}[s_{x_p,y_p} \geq s_{x_q,y_q}]2^{k-1}, \tag{4.6}$$

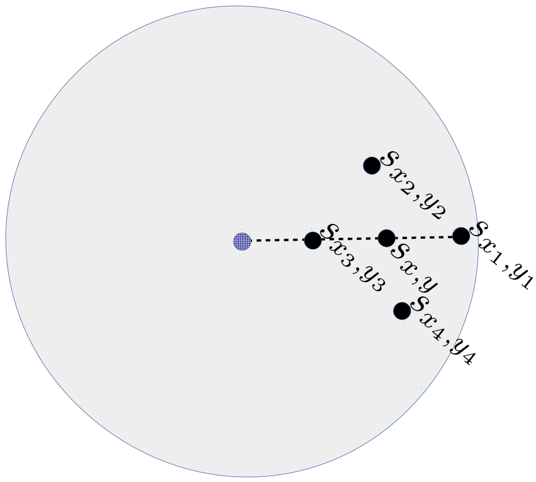
where k is the number of pairs sampled in the patch. $\{x_p, y_p\}$ and $\{x_q, y_q\}$, with $p \neq q$, are two positions inside the patch. Sampling can be done in many different ways, uniform, random, and so on. Since this descriptor is intuitive and simple (yet efficient [CLSF10]) several other BREIF-based descriptors handling several transformations were developed. Some examples are ORB (Oriented rotated BREIF) [RRKB11], FREAK (Fast Retina keypoint) [Ort12], BRISK (Binary Robust Invariant Scalable Keypoints) [LCS11] and others.

2.4.7 Local Intensity Order Pattern

In Local Intensity Order Pattern (LIOP) [WFW11] and its extension [FWH12], given a set of covariant regions detected by methods like Harris- or Hessian-affine, each region is then transformed into a circular patch with predefined radius. Gaussian filtering is used to reduce the noise effect after the region's shape is transformed into a circle. Each patch is quantized into a predefined number of l intensity levels. In each patch, in each level, and at each pixel s_{x_p,y_p}, circular k neighbor pixels are sampled as in LBP. These neighbors are then sorted in a descending order, and each permutation of k is indexed using a lookup table of length $k!$. A histogram of length $k!$ is calculated for each patch. It counts the occurrence of each index, *i.e.* neighbors pattern in the level. The histogram of the patch is a concatenation of the histograms of the levels. It seems that, the use of $k!$ may lead to curse of dimensionality but, in fact, as shown in [WFW11], LIOP is working fine with $k = 4$ and $l = 6$, *i.e.* $4! \times 6 = 144$.



(a) Original Patch.



(b) Rotated.

Figure 4.9: Rotation invariant approach of LIOP. The first neighbor s_{x_1,y_1}, the center of the patch and the pixel of interest $s_{x,y}$ are collinear. This way, the neighbors' pattern (*i.e.* $\{s_{x_1,y_1}, \cdots, s_{x_4,y_4}\}$) keeps the cardinality intact when the patch is rotated, since the collinearity is rotation invariant.

In order to make the neighbors pattern rotation invariant, the center of the patch is used as a reference point (Figure 4.9). For each point $s_{x,y}$, the first neighbor (*i.e.* s_{x_1,y_1}), is the point that is collinear to both $s_{x,y}$ and the center of the patch. Once this property holds for two points in the circular neighborhood of $s_{x,y}$, the farthest point is selected as s_{x_1,y_1}. The neighbors are ordered counterclockwise (Figure 4.9), *i.e.* once s_{x_1,y_1} is identified, s_{x_2,y_2} is in the counterclockwise direction from s_{x_1,y_1}.

2.4.8 Others

Steerable kernels, moments, Gabor kernels, and ridges are also popular approaches to detect and encode features. Those that are related to thin object detection, will be discussed in detail in Chapter 5.

3 Model Building

Building a model consists in capturing the characteristics of classes from training samples [YO12]. These characteristics have to be as unique and descriptive as possible for better generalization. In other words, since features are the fuel for model building, their quality is of major importance to obtain a good model. This step is a machine learning problem, where classification is widely used to build a model. Classification consists in categorizing a set of input samples to a set of classes. In the feature space, a classifier can be regarded as a partitioning function, that, in the optimal case, will produce accurate and representative decision boundaries, which can be linear or non-linear. Classification can be interpreted in many ways, such as, for instance, the likelihood of a sample belongs more to a certain group than to any of the other groups. When there are only two classes, the classification is called binary or binomial, and multi-class otherwise.

In order to classify a sample, one needs to build a model. There are many classifiers available in the literature. To name a few: decision-tree, discriminant analysis, support vector machines (SVM), ensemble classifiers, and nearest neighbor. Due to the success of SVM-methods in binary classification, and once it is regarded as being one of the best [BSI08], it is used in this thesis.

3.1 Support Vector Machines

SVM was invented by Vapnik and Chervonenkis in 1963. However, the SVM algorithm was published the first time in its current implementation by Boser *et* al. [BGV92, CV95]. Since then, it has been used in many classification applications due to its good performance. Perhaps, its intuitive nature made it even more popular. Although SVM is a binary classifier, it can be extended to multi-class classification by applying simple procedures like one-vs-all. The interested reader is refereed to [HL02].

As illustrated in Figure 4.10, SVM seeks a decision boundary (in the simplest 2-D case, a line separating two point clouds in the feature space), such that the distance (functional margin) between this boundary and the nearest data points, in each class, is maximum. These points are called *support vectors*, since they support the discrimination task and, in fact, these vectors will be the only ones associated with weights, as it will be explained later.

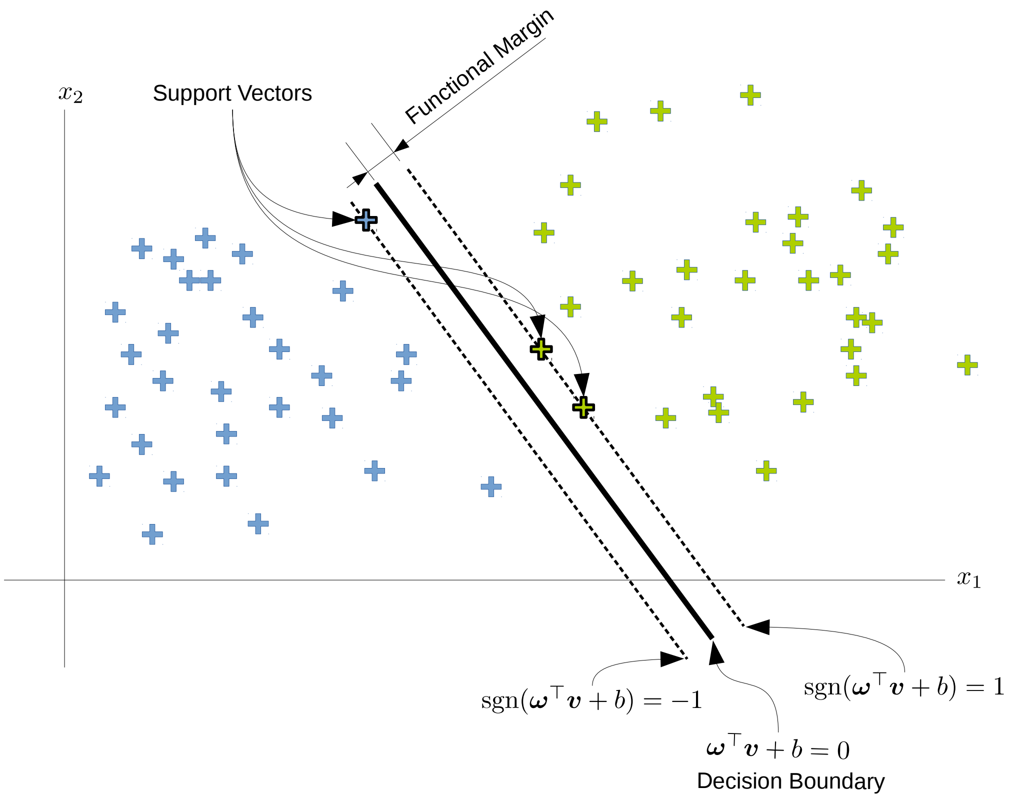


Figure 4.10: SVM classification of two linearly separable classes with two features $\{x_1, x_2\}$.

Given n, d-dimensional, training vectors $\boldsymbol{v}_i \in \mathbb{R}^d$, $i = 1, \cdots, n$, coupled with a labeling vector $\boldsymbol{u}$, the objective of SVM is to learn the mapping $\boldsymbol{v}_i \mapsto \boldsymbol{u}, \forall i$. In other words, SVM seeks a subspace (hyperplane) that divides the ambient space (that surrounds all data points) into two partitions. Assume that $d = 2$, $\boldsymbol{u} \in \{1, -1\}$, and the training data points are linearly separable, the objective is to find a weight vector $\boldsymbol{w}$ that satisfies the following equation [HDO+98, BHW10]:

$$\boldsymbol{w}^\top \boldsymbol{v} = 0 \tag{4.7}$$

This is a hyperplane that can be shifted from the feature space origin by introducing an intercept b:

$$\boldsymbol{w}^\top \boldsymbol{v} + b = 0 \tag{4.8}$$

As a decision function, one can threshold the response of Equation 4.8, such that, whatever is on the one side of the hyperplane is one of two classes, whereas the other class resides on the other side. Therefore, the sign of this equation can be used as a class indicator function:

$$\begin{cases} \text{class 1} & \text{if } \operatorname{sgn}(\boldsymbol{w}^\top \boldsymbol{v} + b) = 1 \\ \text{class 2} & \text{otherwise} \end{cases} \tag{4.9}$$

However, in order to minimize the generalization error, the *discrimination hyperplane* must be set to the best possible position. In order to do that, SVM searches for support vectors that maximize the functional margin. This can be formulated by solving the following quadratic constrained function:

$$\begin{aligned} &\arg\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to:} \quad & u_i(\boldsymbol{w}^\top \boldsymbol{v}_i + b) \geq 1, \forall i \end{aligned} \tag{4.10}$$

where the constraint ensures that the decision boundary is not violated, *i.e.* the training points are classified correctly. By reformulating Equation 4.10 as a *Lagrangian dual problem*, it becomes:

$$\mathcal{L}(\boldsymbol{w}, \boldsymbol{\alpha}, b) = \boldsymbol{w}^\top \boldsymbol{w} + \sum_{i=1}^{n} \alpha_i (1 - u_i(\boldsymbol{w}^\top \boldsymbol{v}_i + b)) \tag{4.11}$$

where α being the Lagrangian multiplier.

Setting $\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = 0$ yields:

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i u_i \boldsymbol{v}_i \tag{4.12}$$

Likewise, $\frac{\partial \mathcal{L}}{\partial b} = 0$ gives:

$$\sum_{i=1}^{n} \alpha_i u_i = 0 \tag{4.13}$$

Expanding Equation 4.11 and replacing $\boldsymbol{w}$ and b, it becomes:

$$\begin{aligned} &\arg\min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j u_i u_j \boldsymbol{v}_i^\top \boldsymbol{v}_j - \sum_{i=1}^{n} \alpha_i \\ &\text{subject to:} \\ &\sum_{i=1}^{n} \alpha_i u_i = 0, \ \alpha_i \geq 0 \end{aligned} \tag{4.14}$$

This has two advantages. First it is only optimizing $\boldsymbol{\alpha}$, and second, it is represented by the dot product of the input vectors, *i.e.* $\boldsymbol{v}_i^\top \boldsymbol{v}_j$, which can lead to a drastic decrease of the computational effort for high dimensional spaces, the following subsection shows this result. Once $\boldsymbol{\alpha}$ is known, the margin can be calculated very easily, and it is obvious to see that, $\boldsymbol{\alpha}$ will be a sparse vector, and has non-zeros only on the support vectors, *i.e.* the margin depends only on the support vectors.

3.1.1 Handling Non-linear Data

Although a linear SVM is a very versatile tool for many problems, using non-linear SVM will be more accurate when proper parameters are found. A non-linear SVM can be used by mapping the data points to a higher dimension using a non-linear function. In fact, such a process is highly computational intensive, especially when solving the optimization problem to find the decision boundary and the margin. A clever way to minimize it would be to bypass calculating high dimensional features and to use the so called "kernel trick". The main idea behind the kernel trick consists in finding a function that can express the dot product of the mapping function without calculating the mapping itself. Suppose that $\phi(v)$ is a non-linear function that maps the feature vector to a higher dimension, the kernel trick is to find a kernel function such that $\mathcal{K}(\boldsymbol{v}_i, \boldsymbol{v}_j) = \phi(\boldsymbol{v}_i)\phi(\boldsymbol{v}_j)$, and replace $\boldsymbol{v}_i^\top \boldsymbol{v}_j$ in Equation 4.14 by $\mathcal{K}$.

3.2 Soft Margin *C*-SVM

For many cases, it is practical to "relax" the margin, such that, a predefined misclassification error is tolerated. Geometrically, this can be interpreted as allowing the margin to contain some learning vectors, as opposed to the *hard margin*, where it is an empty *gutter* (Figure 4.10), this relaxed margin is called *soft margin* [CV95, CL11]. A soft margin kernelized SVM can be written in the matrix form as :

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^\top Q \boldsymbol{\alpha} - \mathbf{1}^\top \boldsymbol{\alpha} \\ \text{subject to:} \quad & \boldsymbol{u}^\top \boldsymbol{\alpha} = 0, \\ & 0 \leq \boldsymbol{\alpha} \leq C \end{aligned} \tag{4.15}$$

where **1** is an all-one vector, and C is a regularization parameter that balances the margin width and the amount of misclassified input vectors. $Q = u_i u_j \mathcal{K}(\boldsymbol{v}_i, \boldsymbol{v}_j)$ is a positive semidefinite matrix with $\mathcal{K}$ being a kernel function (*e.g.* a Gaussian), and $j = 1, \cdots, n$. In fact, Equation 4.15 is equivalent to Equation 4.14 except for the second constraint which is being rebounded using C.

The thresholding function will then be defined as:

$$\operatorname{sgn}\left(\sum_{i=1}^{n} u_i w_i \mathcal{K}(\boldsymbol{v}_i, \boldsymbol{v}^{(\text{test})}) + b\right) \tag{4.16}$$

where $\boldsymbol{v}^{(\text{test})}$ is a test vector to be classified, and b is the intercept.

4 Hypothesis Generation

This section discusses some common approaches for hypothesis generation, it describes the sliding window, voting, segmentation, and some other approaches. For further reading and information, a recent detailed survey can be found in [HBDS15].

It is noticeable in recent approaches that the *Gestalt* principle (*i.e.* the whole is greater than the sum of its parts) is dominant [YO12]. In other words, finding and grouping several discriminative clues is more powerful than solely relying on unrelated parts. This is a major principle found in many methods, especially in the voting methods. For instance, a contour fragment is too weak to classify a shape, whereas grouping multiple fragments, forms a powerful representation. The way of grouping these segments is, however, very challenging due to object detection problems mentioned in Section 1.

Uijlings and *et* al. [UvdSGS13] defined some qualities for a hypothesis generator. A good generator must have:

- Scale-invariant search, where all possible scales of an object have to be considered.
- Merging flexibility, so that, different strategies are used to merge clues.
- Speed.

In addition, other simple criteria, like the repeatability of the hypothesis generator, are suggested in [HBDS15]. The repeatability test, answers the question on how an object generator will perform when an image is being slightly changed *e.g.* by rotation, illumination, or any other factor.

4.1 Sliding Window

Intuitively, the sliding-window technique also known as sub-window search, is the common tool for proposing objects, *e.g.* [RBK96, VJ04]. Proposing an object location is as easy as scanning the image from left to right and top to bottom using a varying size sub-window. Each sub-window is then converted into a feature vector which is validated as containing, or not, an object of a given class. A serious weakness of this technique, however, is that the computational effort required is not acceptable.

In fact, sub-window's search "blindness" nature has been criticized by many authors and they have suggested the development of more efficient search strategies, such as [LBH09]. It has been shown in [GLAM09] that, scanning an image looking for objects has problems beyond computation. Sweeping a window on an image is not an attractive way to locate objects, it is different from segmentation, and it does not take the human visual perception mechanism into consideration, where only salient regions are focused in the scene.

In order to cut down the number of possible sub-windows to be searched, Lampert *et* al. [LBH09] incorporated the branch-and-bound discrete optimization technique, which is widely used to solve problems with massive sizes, such as a dynamic programming problem with a great number of possibilities. Branch-and-bound is a tree generator, where possible solutions are generated in branches (branch). The branches that have no potential solutions are pruned (bound) to reduce the computation effort. In this case, the bounding box will be the solution. Good results are also obtained in [FGMR10] by an improved sliding window search with a part-based HOG descriptor. Edge segments are used in [ZD14] to measure the "objectness". A bounding box is a good candidate if it is likely to contain contours without bisecting the box boundaries. In other words, the box encloses the object, such that no object boundaries intersect the box. A sliding search with intersection over union (IoU) is used to speed up the search. IoU is the ratio of overlap between two bounding boxes. What should also be pinpointed is the fact that IoU is widely used in evaluating the detection accuracy, *e.g.* in the PASCAL Challenge [EGW+10] the IoU ratio is 50%. In [ZD14], when searching for candidate boxes, any move from one position to another, is constrained by the required IoU ratio. This yields a trade-off between the bounding box accuracy and speed.

4.2 Voting

In this technique, features like contours, regions, and so on, are used to create a Hough-like voting scheme. In other words, parts vote for the whole. Using an efficient segmentation approach [AMFM09], Gu *et* al. [GLAM09] developed an approach by creating matching trees of regions between a query and a training sample. Each one of the matching regions will vote for the position and size of the bounding box regarding the ground truth bounding box of the training sample. These votes will then be clustered by mean-shift to minimize the number of boxes to be evaluated. This approach, has the nice advantage to locate objects even with partial region matching, but its grouping strategy has not been validated to discover various parts of objects [UvdSGS13]. Similar approaches can be found in the literature, but with other features like contours [FFJS08]. The adjacent contour segments pairing approach described in [FFJS08], is used to create a *pair of adjacent segments* (PAS) [FJS07], which are line approximations to edges detected by [MFM02]. A PAS is a pair of segments, such that, they are meaningful together, *e.g.* an "L-shape" PAS is composed of two, one vertical and one horizontal, segments. PAS is then represented by a scale invariant descriptor where, for instance, the lengths and the distance between the segments are scale normalized and used as features in the PAS descriptor. Other features like segments' orientations are also used. Each training image is then densely sub-sampled to sub-images, and pairs of adjacent segments are extracted from each sub-image to create a codebook, *i.e.* a bag-of-PAS. This way, each image is described by the set of clusters' centers. This is the same as the general approach described in Appendix B for the SIFT method. An object class model is estimated from the clusters' centers by voting for each PAS location among all training images. In order to find potential locations of the object in a test image, each PAS in the test image is computed, encoded and compared to the PASes in the object class model. Matches between the two sets of PASes are pooled in a 3-D Hough-like matrix to vote for a bounding box and scale of the test sample and therefore, some maximum values of this matrix are chosen. The voting scheme in [RDB10] is similar to the one in [AMFM09] but edge correspondences are used instead of regions with a different ranking approach based on the coverage of the reference contour. In [TMdB15], Kasim *et* al. used the fact that, discriminative contour segments are more likely to belong to one training class than to the other training classes, contour fragments are ranked using a Bayesian framework under the nearest neighbor settings. Shape context is used as a descriptor for normalized fragments. For a hypothesis to be generated, the fragments of the test image are matched to the training fragments. The nearest neighbor training fragment is used to initiate a bounding box hypothesis. This fragment is then used as a trigger to generate more bounding boxes by including more training fragments that are stable, and adjusting the bounding boxes accordingly. A nice property for this approach is the use of a light weight nearest neighbor classification and produce nice results, actually the 3$^{\text{rd}}$ among the state-of-the-art of the ETZH dataset merely using the border information.

4.3 Segmentation

Due to the fact that natural images can be seen as a hierarchy of regions, hierarchical region grouping is used to generate a pyramid of regions, where regions in a level are created by merging regions from a lower level. The pyramid is initiated by super-pixels generated by a segmentation approach [APTB$^+$14, UvdSGS13].

The regions generated by the graph-based segmentation [FH04] are used to create a bottom-up hypothesis generation in an approach called the *Selective Search* [UvdSGS13]. Regions grow (by merging) in a way similar to the general seeded region growing image segmentation approaches [AB94] (see Section 5.5 of Chapter 2) but, in Selective Search, the seeds are super-pixels and the merging criteria is aware of many variations. Neighbor regions are compared and grouped to form bigger regions, until the whole image becomes a single region. The similarity score in this approach is a linear combination of sub-similarity scores gathered from different region neighborhood features, such as color similarity, solidity of the merged region from grouping two smaller ones, texture, and more. In addition, since different strategies are used for region grouping, a ranking technique is used to select which locations are appealing to form an object, and where low ranking locations are ignored. A similar approach can be found in [APTB+14], where a multi-scale hierarchy segmentation using efficiently computed normalized cuts is used to generate a set of candidate objects.

4.4 Other Approaches

Alexe *et* al. [ADF10] defined several measures of objectness, which determines whether or not a window contains a generic object. The measures were calculated in a multi-scale setting to cope with scale variation. These measures are based on generic object features, like boundary, saliency, and appearance distinction from the surroundings. For instance, a superpixel-based measure, which indicates high objectness if the bounding box is not crossing a superpixel, *i.e.* when the whole object is tightly contained by the bounding box. Such measure has shown an interesting distinctiveness and barely fails even when object appearance overlaps with the surroundings. A similar idea can be found in [ZD14], using edges instead of regions. The final score of each sub-window objectness is estimated using a Naive Bayesian Model, which combines the scores from each measure. In fact, this method can also be listed as a segmentation approach because, just as in Selective Search [UvdSGS13], it uses the same segmentation approach to obtain superpixels [FH04]. In [ML11], the edges in a test image are compared and partially matched to a contour model. This matching is used to build a weighed graph where vertices represent the partial matching between edges and contour segments. An edge is created in the graph, if both the query contour segments are near to each other, and the same for the model contour segments that best match them. The weight of an edge in the graph will be the averaged partial matching scores. The hypotheses will then be the maximum cliques in the graph. The objective of using the graph when compared to the voting scheme is to make the hypothesis more robust to background clutter, since edges are created in the graph between two vertices when there are two matches between the query and the model segments. This will reduce the randomness that may occur in the matching, when using a single segment.

5 Hypothesis Verification

Once a sub-window is nominated to contain an object, one needs to answer two questions:

- Does this window contain an object (objectness [ADF10])?
- What is the type of the object (classification)?

For most of the object detection and recognition systems, a single verification is enough to answer both questions (as background samples are included in the training set as negative samples). In most cases, the SVM class indicator function in Equation 4.16 is used. For many other approaches, the Bayesian model [ADF10], and LDA [HMR12] are used.

The average probability of a hypothesis belonging to a class is used in [GLAM09]. Likewise, in [TMdB15] the score of a hypothesis is the average of similarity between shape model's segments and their nearest neighbor segments in the hypotheses. The model's segments are compared against their nearest neighbors in the hypothesis (testing sub-image), to minimize the effect of background clutter. Therefore, clutter fragments in the hypothesis will be ignored and, the incompleteness of the contours will be penalized.

The Thin Plate Spline-Robust Point Matching (TPS-RPM) [CR03] is used in [FJS07] to calculate the score of a hypothesis by sampling the contour of the hypothesis to a 2-D point set. TPS-RPM is a point matching algorithm that transforms a point cloud (test set) into another cloud (reference set). The transformation is iterative under TPS settings, where the cost of the final "morphing" of the test set into the reference set can be used as a good indicator for the similarity between the sets. Up to our knowledge, TPS-RPM is robust to outliers, affine transformations, and a wide range of non-rigid transformations, but it fails in some non-rigid cases [AT12].

6 Conclusions

6.1 Detection of Features

Many of the described feature detectors rely on the local orientation of keypoints to achieve rotation invariance. Scale invariance is achieved mostly by means of scale-space. However, in the problem at hand, microfilariae are featureless objects. In other words, it is not easy to find stable keypoints or, to be more specific, enough keypoints. For instance, considering SIFT, stable local minima or maxima are targeted to be keypoints. In microfilariae, local minima reside mostly in the medial points (the ridge). In Figure 4.6, two keypoints were detected by SIFT. In principle, these two keypoints represent the same feature, *i.e.* a medial axis point. Therefore, it is more suitable to look for *keysequences* (ridges) instead of keypoints. The keysequence can be the medial axis of the microfilaria. For that reason, common keypoint detection approaches are not suitable for the microfilariae problem.

Since covariant-region detectors look for stable regions, they might theoretically be a better choice for this kind of problems. However, in practice, since the detected regions are transformed from ellipses to circles, large regions may become similar to small regions.

It turns out that, color, ridge, and edge features are better candidates. The PAS features discussed in Section 4.2, are not at all suitable for our problem, since microfilariae have a free-form shape and searching for coexistent segments to form PASes is impractical. The same problem exists with other approaches using edges or regions. However, since microfilariae are almost symmetric objects, treating contour features under this assumption might be helpful.

As a final remark, choosing the exact number of training samples with respect to the number of features is, however, a very tedious task. There are two main contradictory assumptions [Lew92]. The first assumption is that, more features result in more information and, hence, better classification. The other one is that, since more features increase the number of hypotheses about the classes, it leads to curse of dimensionality. A rule of thumb for the ratio between the number of training samples and features is, in fact, not easy to achieve. In general, one needs to use as many samples as possible, unless adding more samples starts to decrease the performance.

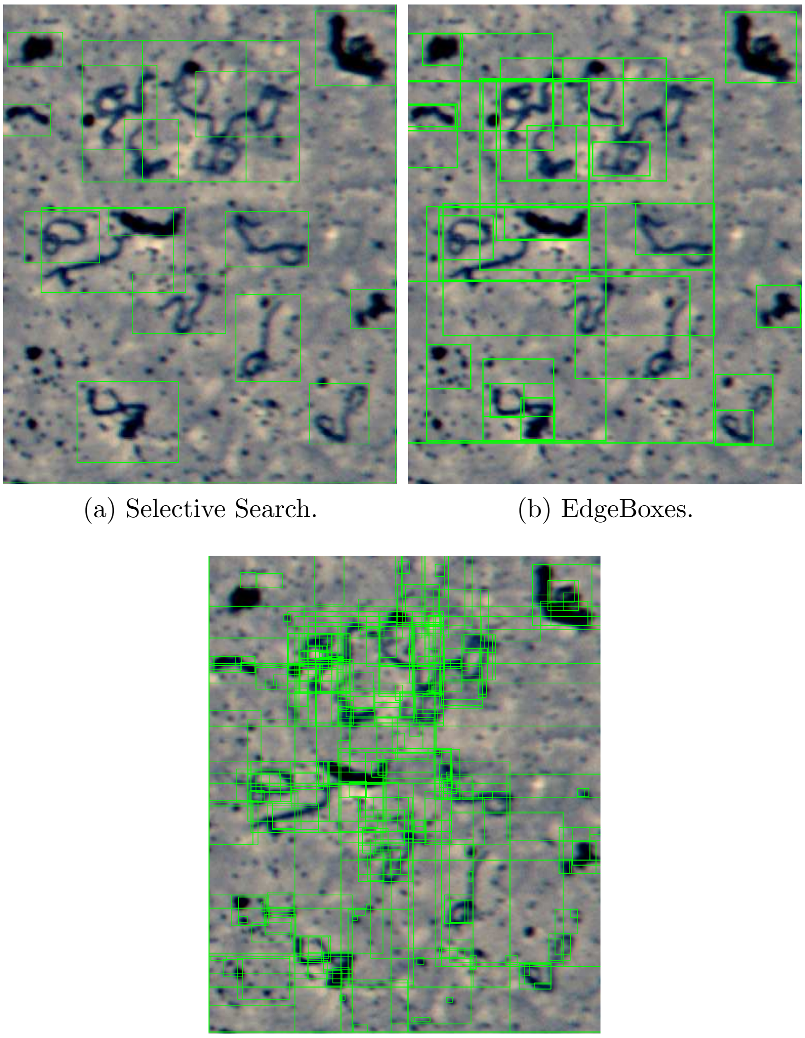
6.2 Model Building

SVM-methods with different kernels and expensive learning paradigms [UvdSGS13] are very powerful model building methods. In fact, it has been shown in [ZVRF12, ZVFR15], that the increase of data size,"effectiveness of big-data", may not be powerful if the model builder is weak. More data needs more sophisticated machinery to capture intra-class variation. In fact, when a huge amount of training data is used, the recall will saturate after a given amount of data and, surprisingly, as demonstrated in the mentioned articles, it drops. Therefore, developing powerful model builders is no less important than creating massive datasets. However, one has to offer the model builder a satisfactory amount of training data, otherwise, the learnt model may not be good enough.

6.3 Hypothesis Generation

Voting and segmentation are appealing choices, even though state-of-the-art methods like [FGMR10], rely on sliding window approaches. It has been shown in [HBDS15] that, Selective Search approach [UvdSGS13], EdgeBoxes [ZD14], and the multi-scale combinatorial grouping (MCG) [APTB$^+$14] are very advantageous when compared to many other approaches. However, since our objects are featureless, and generating parts from the microfilaria is neither intuitive nor appropriate, a different approach is needed. Additionally, background of microfilariae images is supposed to be homogeneous, therefore, a segmentation-based proposal is, perhaps, the best choice for our problem when compared to the discussed approaches, which are developed for natural images. Once Selective Search is designed as a class-independent hypothesis generator, it seems to be a good option for our problem. Our preliminary results showed that, Selective Search is very efficient in terms of computation due to the performance of the segmentation algorithm in [FH04]. In addition, Selective Search produces good object candidates, after some parts of the method were adapted to fit the microfilariae detection problem. Examples of Selective Search, EdgeBoxes, and MCG are shown in Figure 4.11. More details will be discussed in

Section 2 of Chapter 6, when the proposed approach for hypothesis generation is presented.



(a) Selective Search. (b) EdgeBoxes.

(c) MCG.

Figure 4.11: Candidate objects proposed by different approaches.

6.4 Hypothesis Verification

In general, the features used in generating the hypothesis and the machinery used to build the model define the nature of the verification step. In fact, this step can be generic as in [ZD14, ADF10, UvdSGS13], where it verifies the objectness of the detected hypothesis, not the class, such that, incorporating an expensive classifier becomes easier, because many bounding boxes will be pruned from the search space. It has been shown in [ADF10, AF12] that, minimizing the number of non-object candidates, may contribute to recall by minimizing the false positives. Considering the evaluation, when a bounding box is used, the PASCAL IoU > 0.5 [EGW+10] is used to indicate a ground truth hit. Although this measure is used in many state-of-the-art papers, it has been demonstrated in [ZD14] that 0.5 of IoU is weak, and a better overlap is to set IoU ≈ 7.

To summarize, designing an object detector relies, in the first place, on the nature of the object. However, there are common features that can be suitable for many objects, such as edges, but one has to be cautious about background clutter and unwanted artifacts coexisting with the objects. Deformable objects are usually harder to detect, since extracting stable features is more challenging. Microfilaria is one of those objects, where a specialized object descriptor might be necessary to be developed, in order to cope with the microfilariae elasticity. One further important feature of the microfilaria, which was not discussed in this chapter is its thinness. Thinness will be covered in the next chapter.

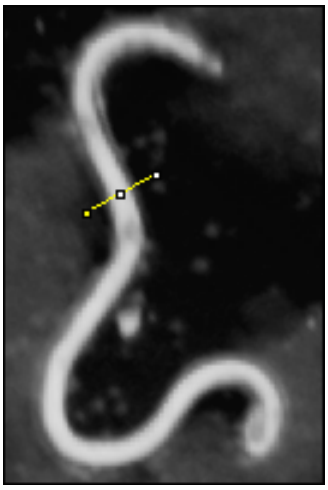
5

THIN STRUCTURE DETECTION

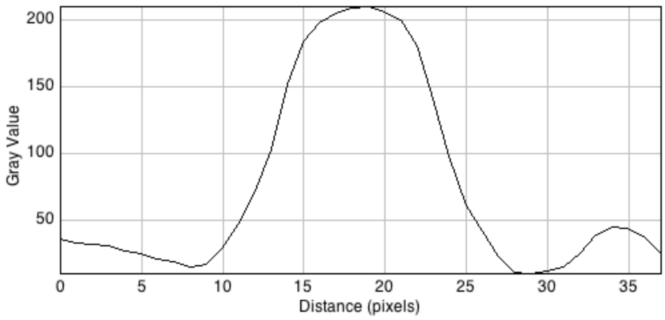
1 Introduction

Microfilariae, vessels, and nematodes share common geometric properties. They are, for instance, thin objects with similar cross-section profiles. However, vessels vary in profile radius within the same image, and a scale-space approach is needed to detect them. In the literature, there are many approaches to detect thin objects in general. This chapter addresses the most common approaches, while a comprehensive review can be found in [KQ04].

Since these thin objects depict Gaussian-like profiles (Figure 5.1), images can be filtered by Gaussian functionals, such as Matched Filters (MF) and Differential Geometry (DG), which will be discussed in Sections 2 and 3, respectively. Other approaches rely on Mathematical Morphology (MM) filtering to highlight thin objects (Section 4). Hybrid approaches are also discussed in Section 5. In addition, thin objects can be traced given some starting initial points as described in Sections 6 and 7. In Section 8, thin objects are described using their local widths.



(a) A microfilaria.



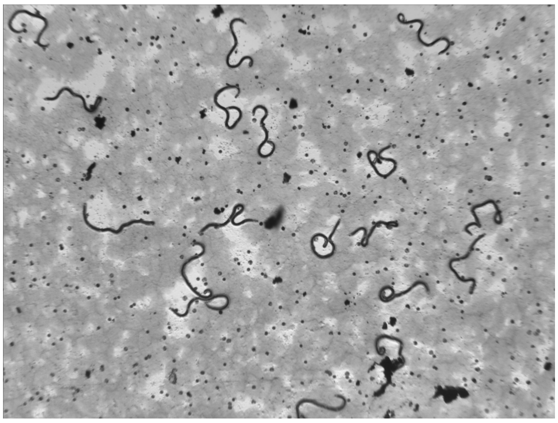
(b) Its cross section profile.

Figure 5.1: The bell-shaped profile of a cross-section of a microfilaria.

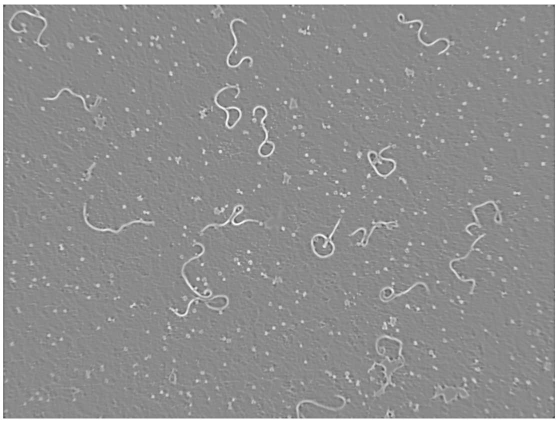
2 Matched Filters

A matched filter is a template that, when correlated with an image, highlights specific features [CCK+89]. In this case, the filter is a 2-D kernel $\boldsymbol{G}$ calculated by sweeping a zero-mean Gaussian function along a direction. Formally, let this direction be y and the length of the sweeping span k, the kernel entry $g_{x,y}$ at position (x, y) is calculated as:

$$g_{x,y} = e^{-\frac{x^2}{\sigma^2}} - \frac{\int_{-\lambda\sigma}^{\lambda\sigma} e^{-\frac{x^2}{\sigma^2}} dx}{2\lambda\sigma} \text{ for } |x| \leq \lambda\sigma, \ |y| \leq \frac{k}{2} \tag{5.1}$$



(a) A sample image.



(b) The response of the MF.

Figure 5.2: The response of the MF ($\lambda = 3.0$, $k = 6.0$, and for 15 rotational steps) applied to a sample image.

where σ, and λ are the standard deviation and the kernel width factor, respectively. In the implementation, two parameters have to be set, the discrete length of the Gaussian after truncation (*i.e.* $\lambda\sigma$, usually $\lambda = 3$ is sufficient) and the rotation step of the kernel, 15 degrees are usually enough [ARQA07].

Figure 5.2b shows the response of the MF applied to a sample image. It is noticeable that, MF works well wherever there is a clear microfilaria, but shows significant sensitivity to objects' edges.

3 Differential Geometry

In this approach, thin objects are detected by examining the response of image derivatives [LA08]. Since thin objects may vary in width in the same image, a scale-space search is used. Therefore, Gaussian derivatives are widely used. As mentioned in Section 2.1 in Chapter 4, the problem of calculating image derivatives can be well posed by convolving the image with the Gaussian derivatives [FtHRKV92], where the Hessian analysis plays the major role in highlighting thin objects.

3.1 Frangi's Multi-scale Filtering

Frangi *et* al. [FNVV98] analyzed the eigenvalues of the Hessian matrix (Equation 2.44) and designed a vesselness measure. In bright vessel-like structure in dark background images, one eigenvalue, λ_1, is expected to be high, while the other one λ_2 is expected to be negative, assuming $|\lambda_1| \leq |\lambda_2|$ ordering (see Section 6 in Chapter 2). The Frangi's vesselness measure is formally defined as:

$$V(s_{x,y}) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \\ e^{-\frac{r^2}{2\alpha^2}}(1 - e^{-\frac{\rho^2}{2\beta^2}}) & \text{otherwise} \end{cases} \tag{5.2}$$

where, $r = \frac{\lambda_1}{\lambda_2}$, is the blobness measure and ρ is the norm of the eigenvalues. The parameters α and β are user defined. For dark objects in bright background, Equation 5.2 has to be changed accordingly.

3.2 Steger's Method

Steger developed a curvilinear detection method based on the Hessian matrix [Ste98]. The image derivatives are calculated by convolving it with the Gaussian derivatives. The centerline is detected by finding the vanishing point of the first derivative, and the eigenvector of the Hessian associated with the largest absolute value, is used to find the direction of the maximum curvature of the cross-section profile. Therefore, this method, can detect both the centerline (ridge) and the boundaries of thin objects. An example of ridge detection using this method is shown in Figure 5.3.

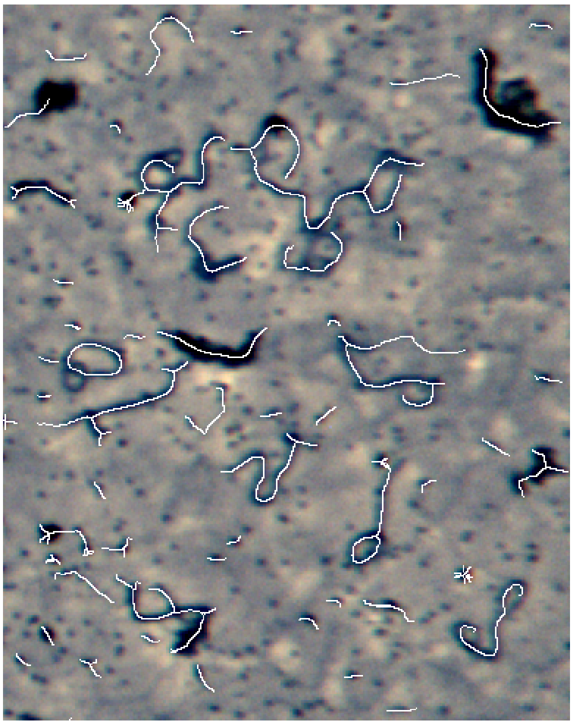


Figure 5.3: Ridges detected by Steger's approach for the sample in Figure 4.11.

4 Local Elevations Detection

Treating the microfilaria image as a topographic surface is another way to study the detection problem. The microfilariae can be seen as a narrow continuous plateau with a constant minima spanning the medial line. However, this assumption is not always the case due to noise and imaging artifacts. Intensity discontinuities in the plateau are also common. Mathematical Morphology (MM) provides several filters to detect elevations in 2-D grayscale images. Only grayscale MM with flat structuring elements (SE) is considered here. A nice property of MM is that, almost all filters, are constructed from two basic operations: dilation and erosion [Ser83]. The results of applying several MM filters to a sample image is shown in Figure 5.4.

The grayscale flat erosion of image $\boldsymbol{S}$ by the structuring element $\boldsymbol{D}$ is defined as:

$$\boldsymbol{S} \ominus \boldsymbol{D} = \min_{(a,b)} \left\{ \boldsymbol{S}(x+a, y+b) \right\} \tag{5.3}$$

where, (a, b) iterates over the domain of $\boldsymbol{D}$. Similarly, the flat dilation is defined as:

$$\boldsymbol{S} \oplus \boldsymbol{D} = \max_{(a,b)} \left\{ \boldsymbol{S}(x+a, y+b) \right\} \tag{5.4}$$

These two operators are very common and easy to implement as they can be conceived as minimum (in case of erosion) and maximum (in case of dilation) filters. Inside a region defined by the geometry of $\boldsymbol{D}$ (*e.g.* a disk), each pixel of the image is replaced by either the maximum or the minimum neighbor according to the operator type.

Using these two basic operators, additional filters can be built. Two other popular operators are *open* and *close*. Opening is defined as the erosion followed by dilation:

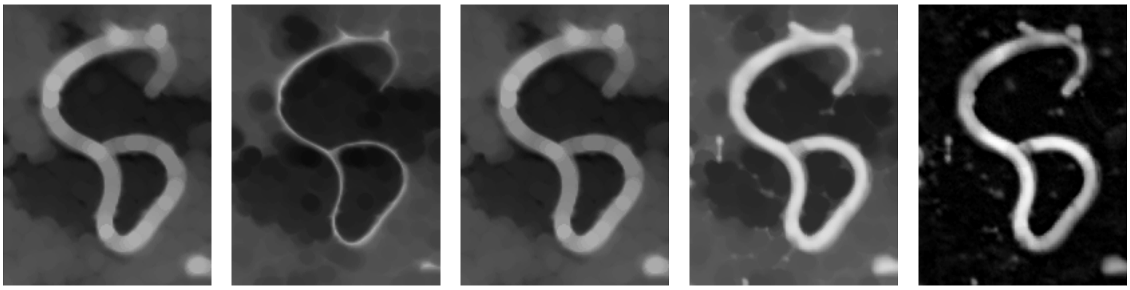
$$\boldsymbol{S} \circ \boldsymbol{D} = (\boldsymbol{S} \ominus \boldsymbol{D}) \oplus \boldsymbol{D} \tag{5.5}$$

Likewise, mathematical closing is defined as dilation followed by erosion.

$$\boldsymbol{S} \bullet \boldsymbol{D} = (\boldsymbol{S} \oplus \boldsymbol{D}) \ominus \boldsymbol{D} \tag{5.6}$$

Two specialized operators can be used to detect elevations: top-hat and bottom-hat transforms. A top-hat transform is defined as the difference between the image and an opened version of itself.

(a) Sample image.



(b) Dilation. (c) Erosion. (d) Opening. (e) Closing. (f) Top-hat.

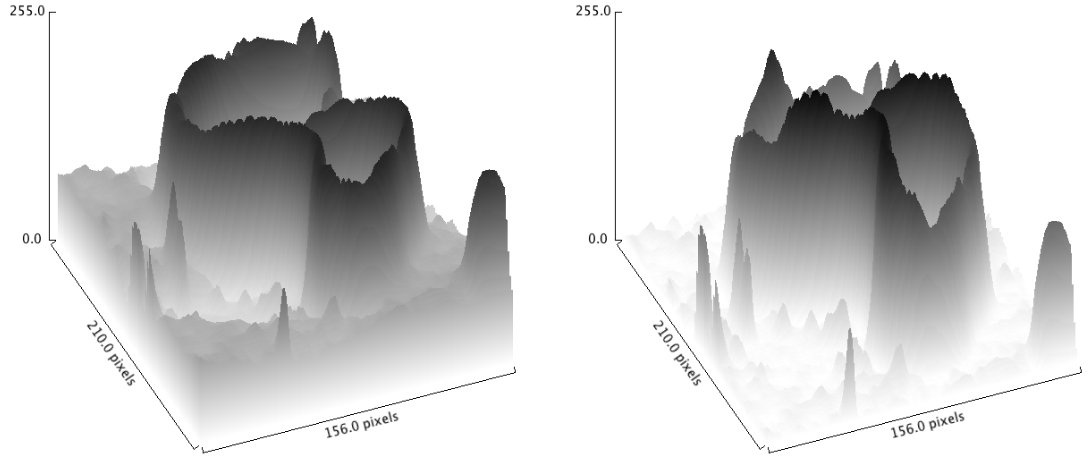
Figure 5.4: The responses of several mathematical morphology operators applied to a sample image using a disk structuring element $\boldsymbol{D}$ of radius 5.

$$\tau^{(\mathrm{top})}(\boldsymbol{S}, \boldsymbol{D}) = \boldsymbol{S} - (\boldsymbol{S} \circ \boldsymbol{D}) \tag{5.7}$$

The bottom-hat transform is the difference between the image and a closed version of itself.

$$\tau^{(\mathrm{bottom})}(\boldsymbol{S}, \boldsymbol{D}) = \boldsymbol{S} - (\boldsymbol{S} \bullet \boldsymbol{D}) \tag{5.8}$$

The top/bottom-hat transforms were used in many vessel-like structure detection methods *e.g.* [ZK01, MC06], due to the useful property they have in highlighting specific features. In Figure 5.4f, the top-hat highlights the microfilaria and reduces the background artifacts (see surface plot in Figure 5.5).



(a) The surface of Figure 5.4a. (b) Result of applying the top-hat.

Figure 5.5: The surface plot of the top-hat transform applied to the sample in Figure 5.4a.

5 Oriented Morphology

Straight structures can be detected by using oriented line structure elements [YYH93, ST01], nevertheless, straight structure elements are not effective for curvy objects. Therefore, other approaches were developed. To name a few, flexible linear openings and closings [BT00], path opening and closing [HBT05, HBT04], parsimonious path openings and closings [MDD14], and Morpho-Hessian [TTD08].

5.1 Path Opening

Path opening (PO) is an adaptation of global path finding between two points, which is a classical problem, usually solved using the Dijkstra algorithm. In local path finding, there is no information about the start and end points. Knowing the length of the path helps to search for such path. Given a 2-D grid represented by a directed graph with a given connectivity scheme, *e.g.* 4-connectivity, the dilation at a node is the set of its successors. Likewise, the erosion of a node is the set of its predecessors. Path opening of length l is, then, the union of all possible paths in the grid of length l. In this case, path opening looks for elongated objects with a specific length. Due to discontinuities in the thin objects, path opening can search for paths even if there is a number of missing pixels along the path. An example of path opening is shown in Figure 5.7e, further ahead.

5.2 Morpho-Hessian

This approach is a combination of both differential geometry and mathematical morphology [TTD08]. The orientation at a point of a thin object is calculated from the Hessian matrix (Section 6 in Chapter 2) and then, orientation is used to guide the morphological filtering process. This is very helpful to apply anisotropic structuring elements (*e.g.* ellipse instead of disk), since the local direction of a thin object will have the same direction as that of the major axis of the anisotropic structure element itself. The object descriptor developed in Chapter 6 is inspired by this approach to estimate the microfilaria local orientation.

6 Path Finding

In [FL95], the user selects the vessel by drawing two lines at the two ends of the vessel. The area between these two lines is sectioned evenly to a set of cross-sections. In each cross-section the medial line and the borders of the vessel are found using MM filters.

The 1-D $\tau^{(\text{top})}$ operator is used to detect the candidates of the medial line of the vessel. The dilation gradient $\nabla_{\oplus}$ is used in each cross-section to detect the border candidates. $\nabla_{\oplus}$ is defined as the difference between the image and its dilated version:

$$\nabla_{\oplus}(\boldsymbol{S}, \boldsymbol{D}) = \boldsymbol{S} - (\boldsymbol{S} \oplus \boldsymbol{D}) \tag{5.9}$$

Among the medial line candidate samples detected by $\tau^{(\mathrm{top})}$, a simplified Bellman–Ford algorithm is used to detect the lowest cost path from the source (at the first line selected by the user) to the sink (at the last line selected by the user). The borders at each side of the detected medial line are also found using Bellman-Ford's.

This approach has two limitations. First, the manual selection of each vessel is a laborious task. Second, the lowest path finding assumes no overlap of the vessels which is not always the case. On the other hand, this approach presents a good scheme to develop a robust vessel detection algorithm given that $\nabla_{\oplus}$ shows good response for vessel-like objects. The $\tau^{(\mathrm{top})}$ can also be used due to its nice response to the vessels' medial axis.

Similarly, given user-defined start and end points of the vessel, [MJS+04] developed a neurite tracing method where vessel detection is based on the response of the Gaussian derivatives. A cost function is formulated based on the eigenvalues of the Hessian at each pixel, and the path is found by the Dijkstra path finding algorithm [Dij59].

7 Tracking

In [YCS00], a gray scale thinning approach is proposed. The vessel under study is selected by a root (seed) point and a point at the tip/end of each ramification. An ordered region grown (ORG) graph is then built to represent the image as an acyclic graph and it is constructed by tracking the expansion of a region. The region grows from its border following the pixel of the maximum intensity. A pruning approach is then applied to remove the least significant branches. Any branch and its descendants are removed if its length is less than a threshold and has a sibling longer than itself. A pyramid of graphs is proposed in [SHDB89] for thin structure detection.

8 Binary Descriptor

Since thin objects manifest uniform thickness, *i.e.* the width of the object has small variations, and particularly, in microfilaria, this variation can only be noticed at the two ends, the local width of the microfilaria can be used to design a descriptor. For that purpose, descriptors like the generalized cylinder descriptor [AB76] or the quench function can be used. A bin h_k in the histogram $\boldsymbol{h}$ of a binary object P, with medial axis $\zeta(P)$ and distance map $\psi(P)$ (Section 7 of Chapter 2) can be calculated as:

$$h_k = \sum [\psi(\boldsymbol{p}) = k], \quad \forall \boldsymbol{p} \in \zeta(P) \text{ and } k = 1, \ldots, \max_{\forall \boldsymbol{p} \in \zeta(P)} \psi(\boldsymbol{p}) \tag{5.10}$$

where $[\square]$ is the Iverson bracket.

In Equation 5.10, an object is represented by the number of maximal disks it has and by their sizes. Figure 5.6 shows synthetic objects and their corresponding histograms. An ideal microfilaria-like object (top row) produces almost a single response (sparse histogram). In other words, let the support of $\boldsymbol{h}$ be $\rho(\boldsymbol{h}) = \{1 \leq k \leq \max_{\forall \boldsymbol{p} \in \zeta(P)} \psi(\boldsymbol{p}) | h_k \neq 0\}$, the cardinality $\mathrm{Card}(\rho(\boldsymbol{h})) \ll \max_{\forall \boldsymbol{p} \in \zeta(P)} \psi(\boldsymbol{p})$ [SMF10]. In this histogram, the "energy" is concentrated at a single bin because a microfilaria is supposed to be composed of a single local structure.

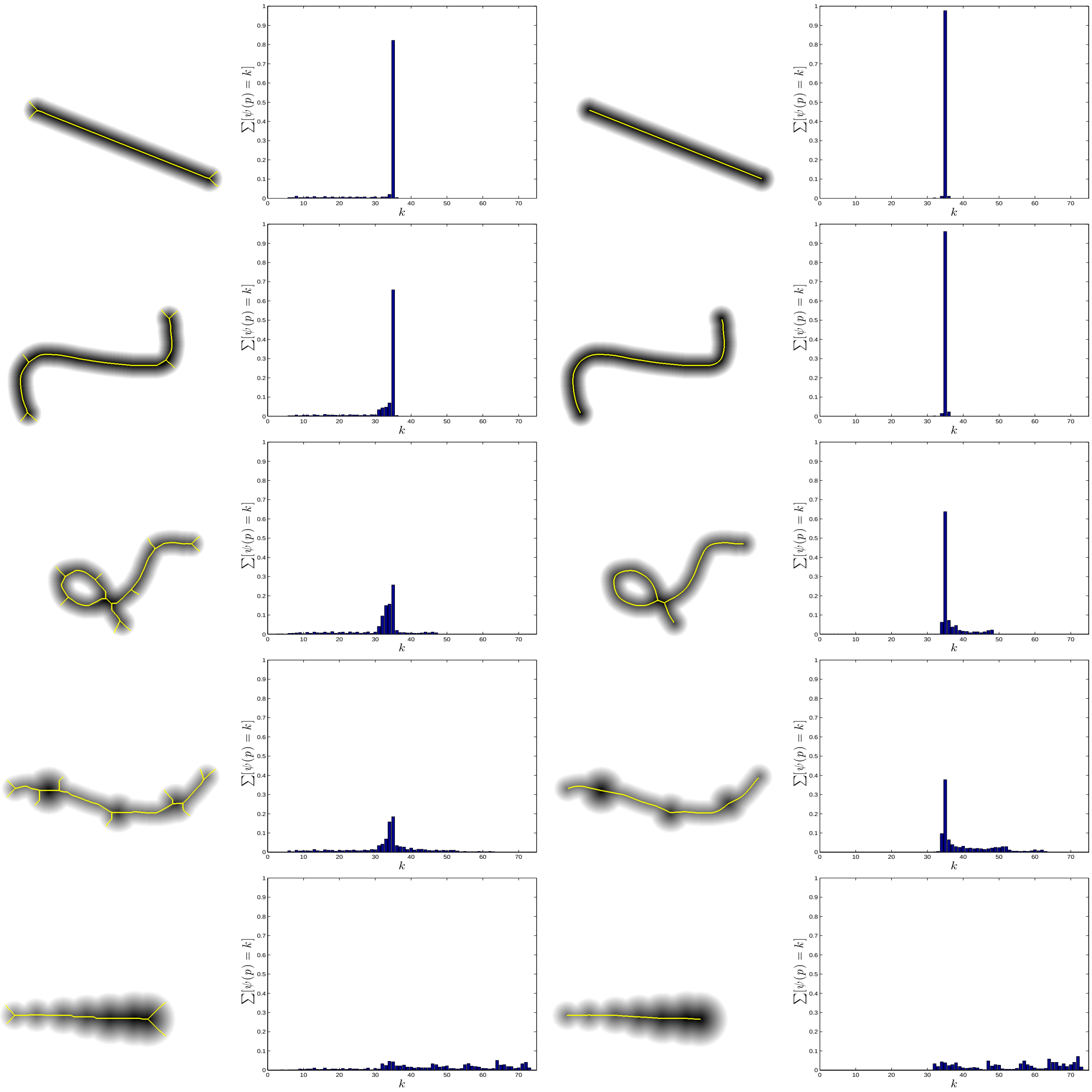


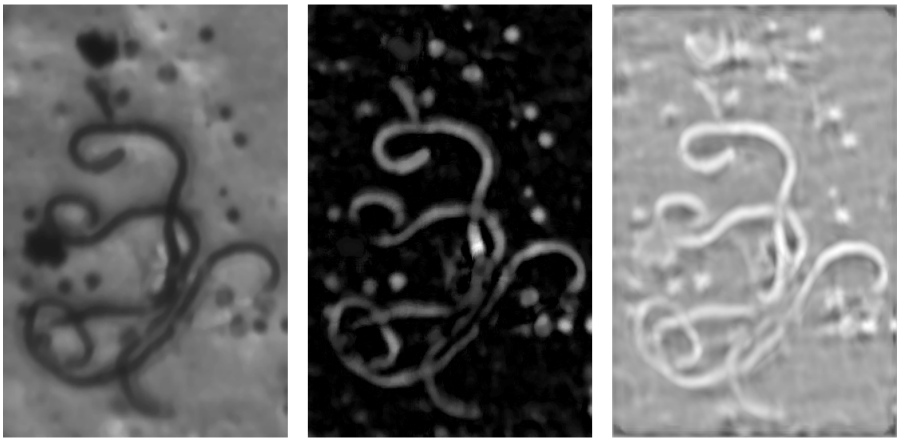
Figure 5.6: Histogram representation of synthetic objects. From left column to right: medial axis superimposed on the EDM of the objects; histogram of the EDM; medial axis after pruning unwanted juts by using [TVW02]; and histogram of EDMs after pruning the skeleton.

When microfilaria bends or overlaps itself or other particles, a good amount of energy will still be available in the histogram representing the microfilaria. Since noise in the contour will produce noisy skeleton (Figure 5.6), the approach presented in [TVW02] is used to prune the noisy juts from the skeleton. The pruning of skeleton produces better histograms. For instance, in the third row in Figure 5.6, one can observe an overlapped object whose histogram is still similar to the ideal histogram presented for the first example, in spite of the overlapping. This makes this representation helpful for featureless objects under non-rigid deformations.

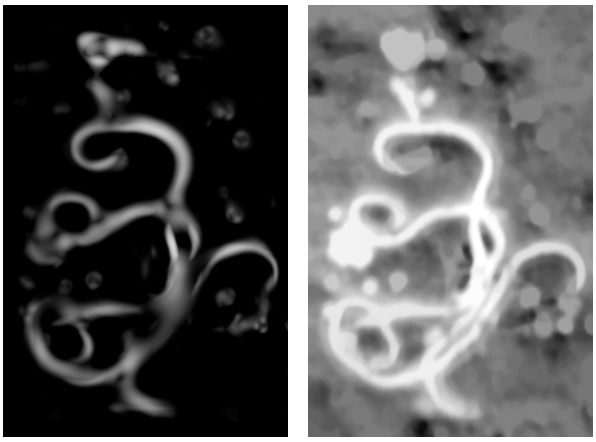
The sparsity of this descriptor, however, presents a problem once it makes the concatenation with other feature vectors impractical. A solution to this problem can be the use of ensembling multi-classifiers [ATdAP+16]. In addition, this descriptor is not rich, because it does not take the local width variation into consideration, and the spatial information is lost in this kind of representation.

9 Conclusion

The results of filtering of the microfilariae in Figure 5.7 can be used to understand the responses of the described approaches. In MM, the shape and size of the structuring element are important to obtain the required results. Due to the nature of the profile of microfilariae (Figure 5.1), the disk structuring element is appropriate for detection. However, in case of microfilariae overlap, the response of the top-hat transform may yield gaps between the microfilariae segments. Additionally, the parts of the microfilariae that are hidden by blood artifacts are not detected (Figure 5.7b). This can be solved by increasing the disk size. Although MF and DG can easily be adapted to scale-space in order to enhance thin objects, they may highlight edges as well (Figures 5.7c and 5.7d) . In contrast, MM is simpler and more appropriate to enhance the microfilariae, due to the fact that, unlike vessels, all microfilariae in the same image have very similar widths.



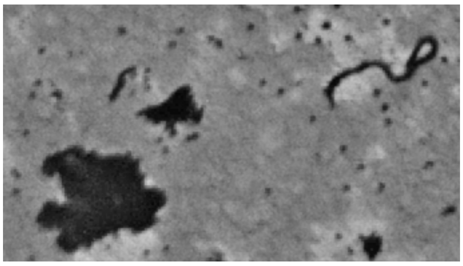
(a) A sample. (b) Bottom-hat. (c) MF [ZZZK10].



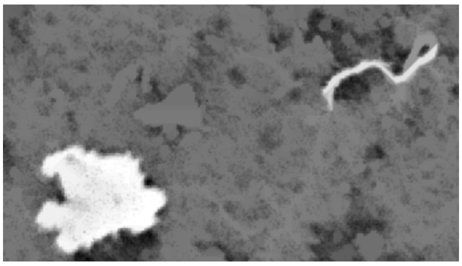
(d) Frangi [FNVV98]. (e) PO [HBT05].

Figure 5.7: The response of different detectors on overlapped microfilariae.

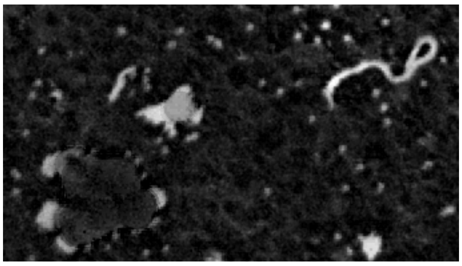
Oriented morphology is very promising (Figure 5.7e), but some problems were experienced with path opening. Figure 5.8 shows the results of path opening and MM. Path opening may produce partial enhancement and detect large objects, which can be avoided by MM. Morpho-Hessian has the advantage of using information both from the DG and MM. Other tracking methods, require user selection of two points for each object, which is not practical for the current problem.



(a) A sample.



(b) PO.



(c) Bottom-hat.

Figure 5.8: A microfilaria sample with a large artifact.

6

MICROFILARIA DETECTION WITH THE HESSIAN-POLAR CONTEXT

"The trick to forgetting the big picture is to look at everything close-up."

– Chuck Palahniuk, American novelist.

1 Introduction

It is difficult to model microfilaria in spite of its simple shape because of its elasticity. Nevertheless, after reviewing common approaches in generic and thin object detection in Chapters 4 and 5, it becomes clear that, microfilaria is better described locally, *i.e.* using a local structure. Therefore, this problem can be studied under the following assumption: *microfilaria is a thin local structure spanning a path.* In other words, sweeping a local structure along a path will yield a microfilaria. In fact, this assumption is a bit rough since, at the two ends of the microfilaria, the local structure radius becomes smaller, and when microfilariae overlap, the structure radius becomes larger. Therefore, in order to describe the microfilaria, these three aspects have to be handled:

- The medial path spanning the microfilaria.
- The shape and size of the local structure at each medial point.
- The local orientation at each medial point.

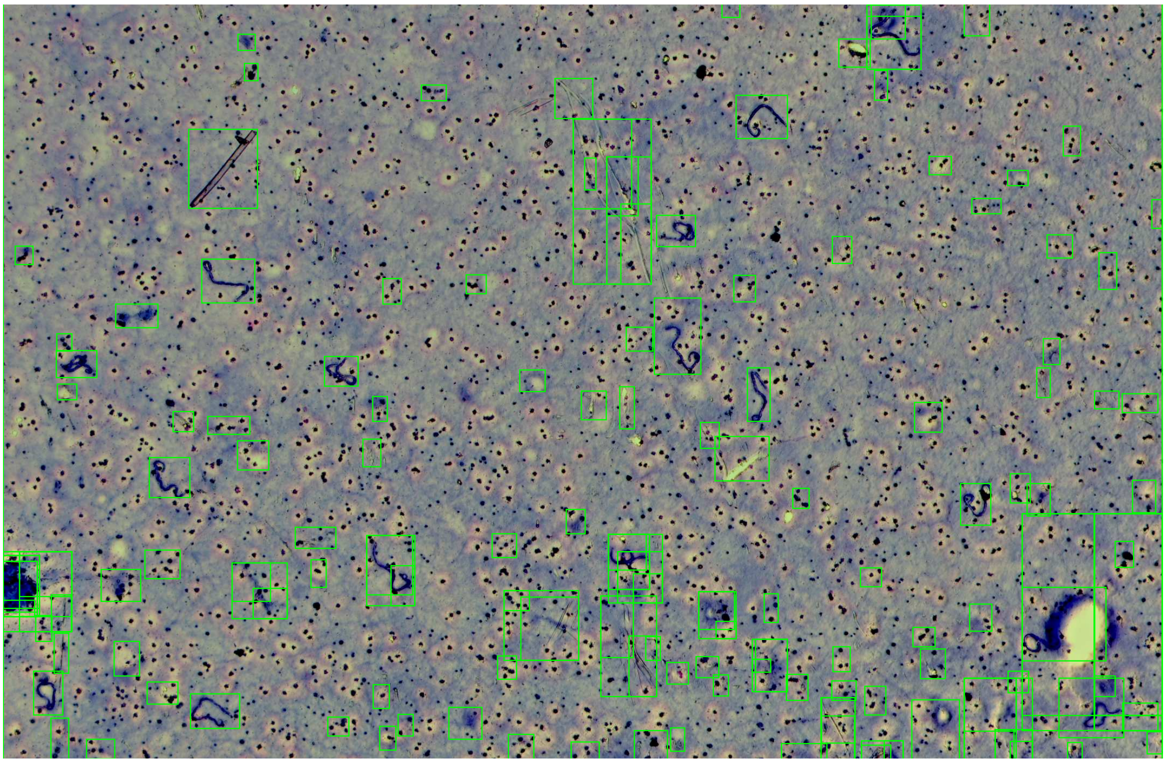
This chapter presents one of the main contributions of this thesis, the Hessian-Polar Context (HPC). In addition, throughout this chapter, important parts of previous chapters will be discussed and evaluated. The hypothesis generation approach is presented in the next section. Microfilaria local width and orientation estimation will be addressed in Section 3. Evaluation of the HPC will be presented in Section 4, and the chapter is concluded in Section 5.

2 Hypothesis Generation

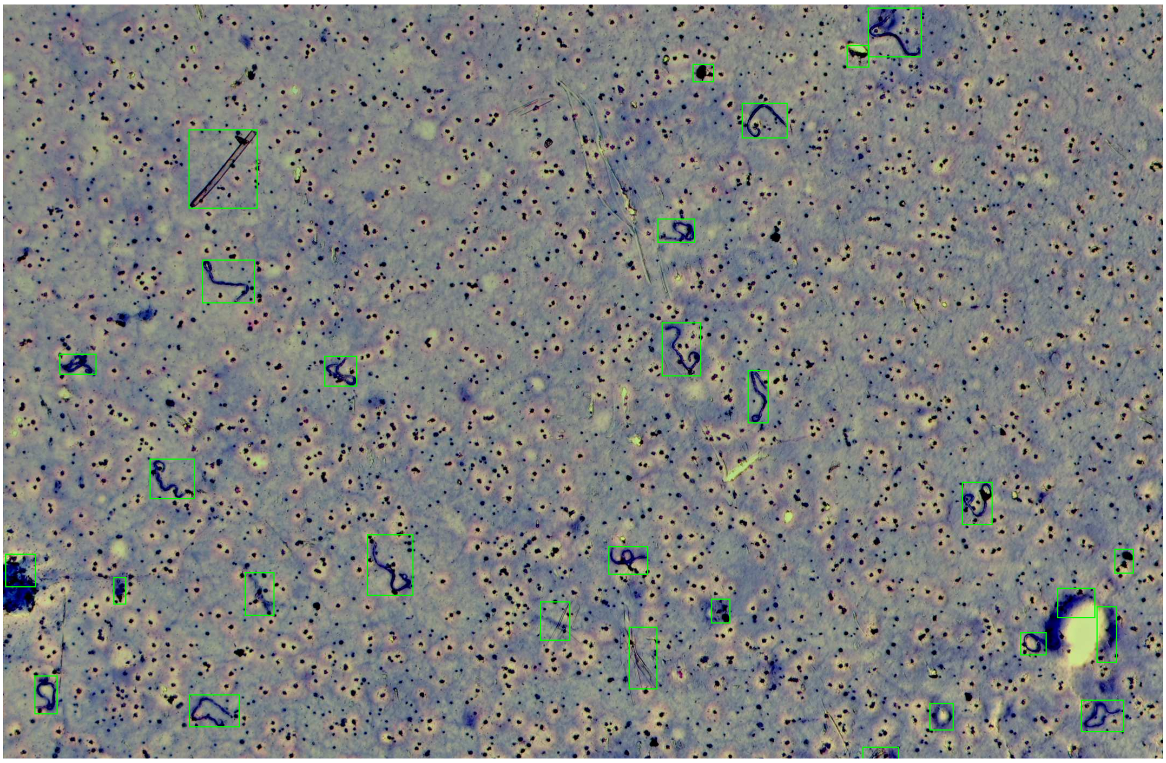
In Section 4 of Chapter 4, several hypothesis generators were discussed. It has been shown that, due to the fact that the background of the microfilaria images is homogeneous, segmentation approaches are good options when compared to many state-of-the-art approaches (Figure 4.11). This section further investigates the effectiveness of the Selective Search [UvdSGS13] and proposes a simple hypothesis generator.

In Selective Search, several sub-measures are used to merge regions generated by segmentation. These measures include texture, color, and size similarity. In addition, merged regions are constrained to be uniform. The uniformity here is meant to avoid forming unwanted regions, by preventing two regions to merge if their contact zones are small. However, when dealing with microfilaria, this sub-measure has to be neglected, because if two regions belong to the same microfilaria, they might have small contact zones due to their shape nature. Besides, the texture similarity measure is also unused. This can easily be adapted by zeroing the weights of these two measures in the linear combination of the measures (Equation 6 in [UvdSGS13]). As stated earlier, besides being effective in producing meaningful candidates, Selective Search is also very efficient in computation demands. Nevertheless, Selective Search produces some unwanted candidates, but these can easily be avoided (Figure 6.1a). This is not a shortcoming of the Selective Search as it is designed to detect possible pronounced regions in natural images, which have complex backgrounds. However, in microfilaria images, these unwanted regions might hinder the recognition task and produce false positives.

In order to produce leaner candidates, and as an alternative to Selective Search, first one needs to suppress as much unwanted artifacts as possible, without sacrificing the quality of microfilariae appearance and, only then, apply a segmentation approach. To achieve that goal, tophat-like transforms can be used to highlight thin objects and reduce unwanted artifacts (Figure 5.7b). The illumination correction approach proposed in Chapter 3 is applied to the input grayscale image, which is then bottomhat-transformed to highlight the microfilariae. Thereafter, the simple mean-c segmentation method presented in [PW96], and described in Section 5.1.2 in Chapter 2, is applied to obtain a binary image, $\boldsymbol{S}^{(\text{binary})}$. The results of this method are shown in Figure 6.1b. It is clear that, this method, is quite satisfactory, because



(a) Selective Search [UvdSGS13] .



(b) Proposed approach.

Figure 6.1: Object candidates produced by Selective Search and the proposed approach.

it avoids many unwanted objects proposed by Selective Search.

Although Selective Search is the state-of-the-art, the proposed simple approach produces better candidates for the microfilaria problem (Figure 6.1).

3 Hessian-Polar Context Descriptor

A new descriptor that we named Hessian-Polar Context (HPC) is now presented. It is a local descriptor calculated at each medial point of the microfilaria. Figure 6.2 depicts the steps of HPC. To start off, the medial axis of the microfilaria is detected, producing medial points and, then, the local radius of the microfilaria is estimated. The orientation of each medial point is calculated and, finally, a descriptor is calculated at each point, producing a feature vector for the microfilaria.

3.1 Medial Points' Detection

Medial points can be detected from ridges, by methods such as Lindeberg's [Lin96] or Steger's [Ste98]. Figure 5.3, shows that, when these methods are used, background clutter produces unwanted ridges. In addition, since the binary components are already obtained from the hypothesis generation developed in Section 2, the thinning of the binary objects seems to be a better option (see Section 7.3 in Chapter 2). Comparing the Medial Axis (MA) detected by Telea in Figure 2.21 and the ridges detected in Figure 5.3, it is clear that Telea's is a better option because it minimizes the number of unwanted extensions in the microfilariae.

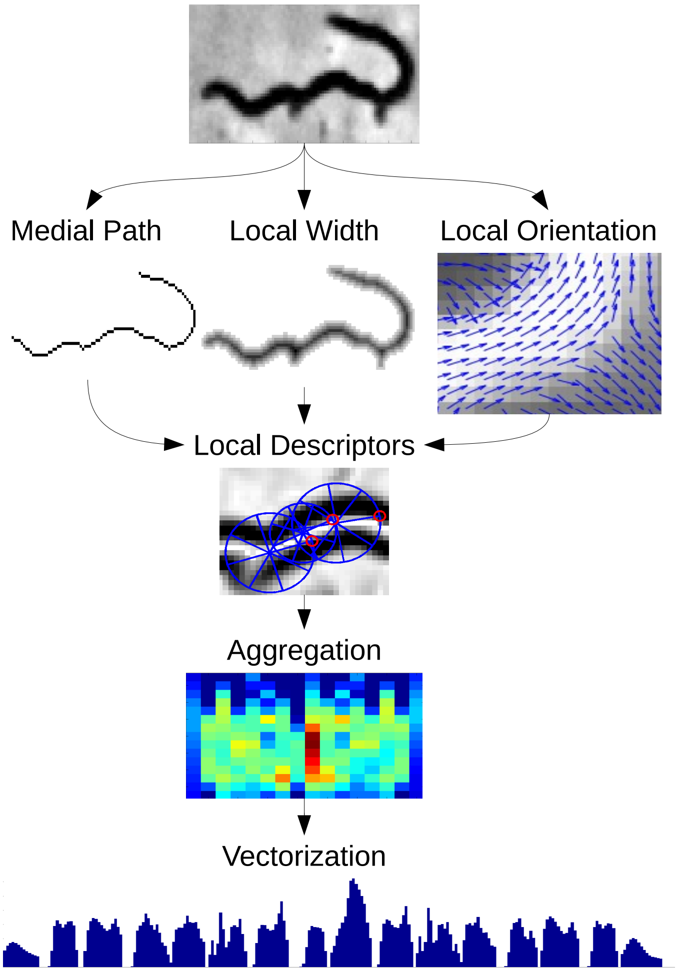


Figure 6.2: Calculating the Hessian-Polar Context descriptor.

3.2 Local Radius Estimation

The profile of thin objects can be modeled using a Gaussian (Figure 5.1), as explained in Chapter 5. In addition, Gaussian derivatives are used to detect the pixels' *thinness* in the images, where the Hessian matrix has shown to be very useful in revealing the structure of the pixels' vicinity. In addition, due to the nice property of the Gaussian to facilitate scale-space, several multi-scale approaches have been developed to detect the local radius of thin objects. Among them, Frangi's approach (Section 3.1 of Chapter 5) was selected and applied in this work due to two main reasons. First, its parameters are easy to set, and second, its response can be used in HPC to estimate the local orientation, as used in the Morpho-Hessian [TTD08]. Since Frangi picks the maximum response among ever-increasing scales σ of the Gaussian derivatives, one needs to estimate the width of the microfilaria in pixels. This can be done by trimming the Gaussian at three standard deviations from the mean.

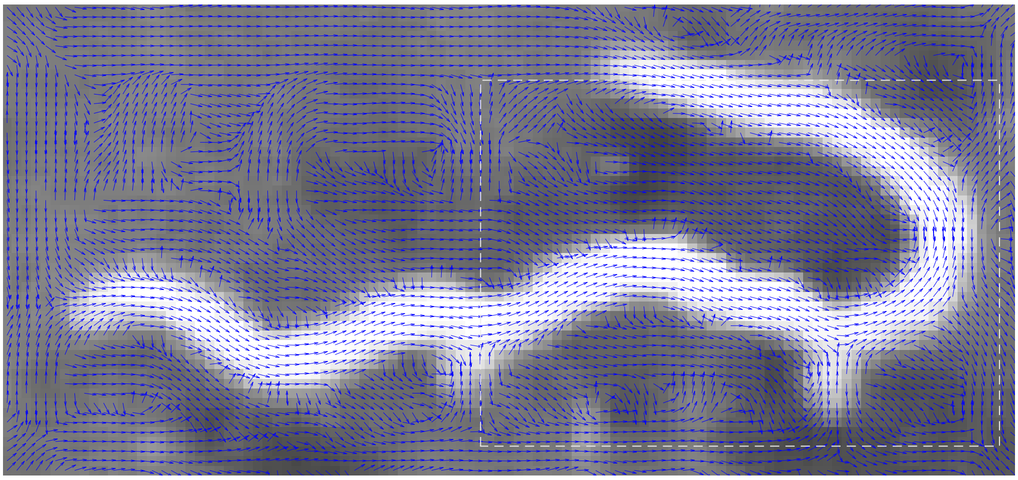
In addition to the radius detection by Frangi's, since the binary connected components have already been detected at this point, estimating the radius directly from the Euclidean Distance Map (EDM) is a faster option. Therefore, although both approaches provide good estimations, EDM was chosen in this work.

3.3 Orientation Estimation

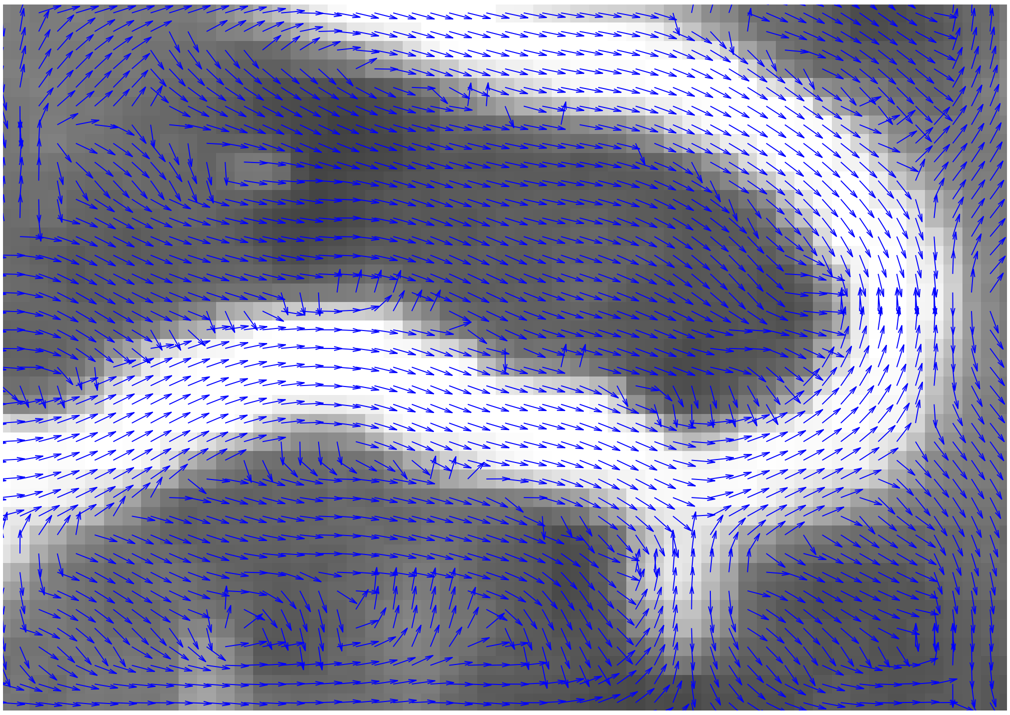
In SIFT-like methods, orientation is chosen based on a histogram voting scheme, where multiple orientations can be assigned to a single keypoint. In fact, this step was criticized in [FWH12], and an implicit reference angle is assumed to be a better option. In HPC, orientation is estimated in a similar way as in the Morpho-Hessian approach described in Section 5.2 of Chapter 5, where it has been shown to be accurate. An example of local orientation estimation is shown in Figure 6.3. When Frangi's is used to estimate the radius, the orientation can easily be obtained but, when the EDM is applied to estimate the radius of the local points, the scale of the Gaussian can be estimated from the EDM.

3.4 Implementation of HPC

In order to calculate the HPC, the Hessian matrix $\mathcal{H}$ is calculated by convolving $\boldsymbol{S}$ with the Gaussian second order derivatives as shown in Section 6 in Chapter 2. Assuming absolute ordering, the orientation of the eigenvector associated to the



(a) The orientation of the eigenvector associated with the smallest absolute eigenvalue.

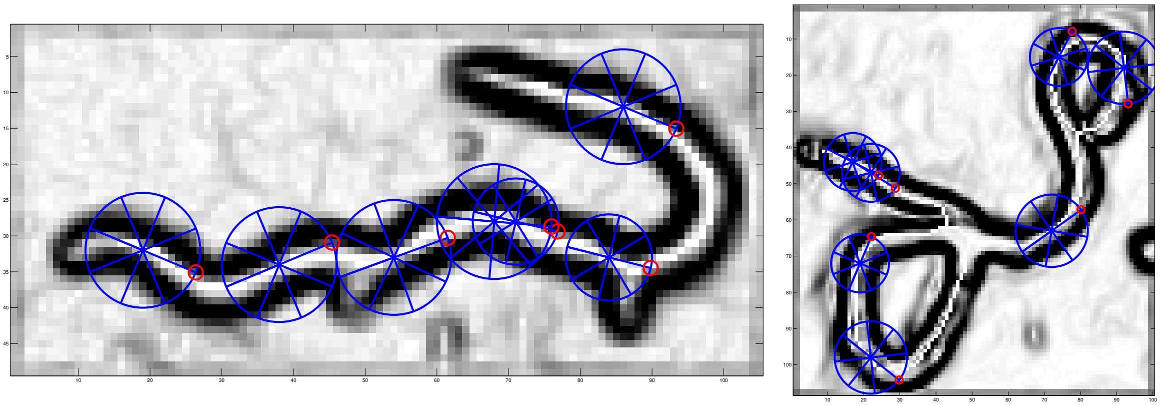


(b) Zoom of the selected area.

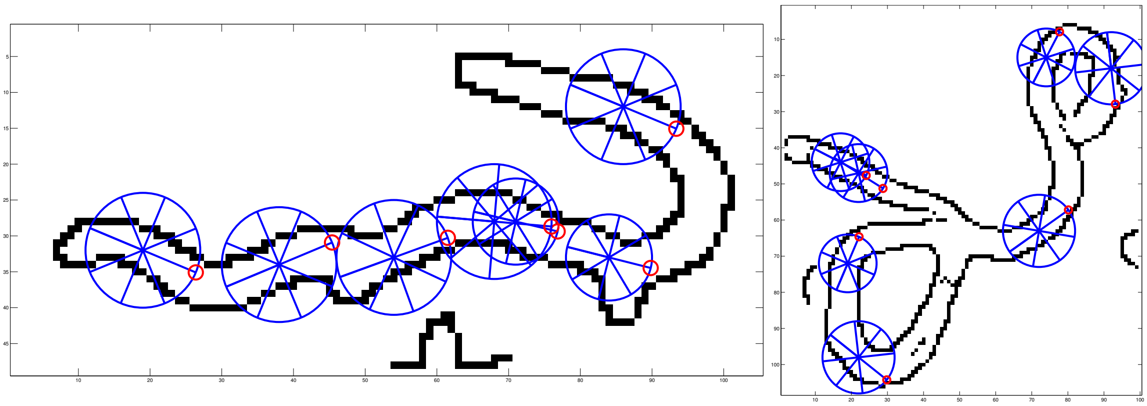
Figure 6.3: The orientation field of the eigenvector associated with the smallest eigenvalue superimposed on the sample of Figure 4.3a. The Hessian matrix was calculated for $\sigma = 5$. The sample is inverted for better visualization.

smallest absolute eigenvalue will be denoted by $\theta \in [-\pi, \pi]$. The orientation field of this eigenvector is shown in Figure 6.3.

At a medial axis point $\boldsymbol{p} \in \zeta(P)$, where $\zeta(P) \subseteq P$ is the set of medial points of object P, a matrix $\boldsymbol{D_p}$ of size $r \times k$ $(r, k \in \mathbb{Z})$ is created to describe the microfilaria at that point. The radius of a circle estimated from the EDM is r and it is centered around $\boldsymbol{p}$, and k is the number of sectors in the circle, as shown in Figure 6.4.



(a) HPC superimposed on the gradient magnitude.



(b) HPC superimposed on the edges.

Figure 6.4: The HPC calculated for some points of two samples. The large blue circles are the HPC segmented into 8 segments and the red circles are the canonical angles of the HPC descriptor.

In order to calculate $\boldsymbol{D_p}$, let $N_r(\boldsymbol{p}) = \{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_l\}$ be the set of neighbors of $\boldsymbol{p}$, within a circular region of radius r. The following steps are repeated $\forall \boldsymbol{q} \in N_r(\boldsymbol{p})$,

1. A rotated version $\hat{\boldsymbol{q}}$ of $\boldsymbol{q}$ is obtained:

$$\hat{\boldsymbol{q}} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \boldsymbol{q} \tag{6.1}$$

2. Point $\boldsymbol{q}$ is normalized with respect to its distance to $\boldsymbol{p}$, and transformed from Cartesian to Polar coordinates:

$$r_{\boldsymbol{q}} = \sqrt{q_x^2 + q_y^2} \tag{6.2}$$

$$\theta_{\boldsymbol{q}} = \tan^{-1}\left(\frac{q_y}{q_x}\right), \tag{6.3}$$

   where $\theta_{\boldsymbol{q}}$ is quantized to the range $[0, k]$.

3. The element $d_{r_{\boldsymbol{q}},\theta_{\boldsymbol{q}}}$ in $\boldsymbol{D_p}$ at position $\{r_{\boldsymbol{q}}, \theta_{\boldsymbol{q}}\}$ is calculated as:

$$d_{r_{\boldsymbol{q}},\theta_{\boldsymbol{q}}} = |\nabla \boldsymbol{S}_{\hat{\boldsymbol{q}}}|, \tag{6.4}$$

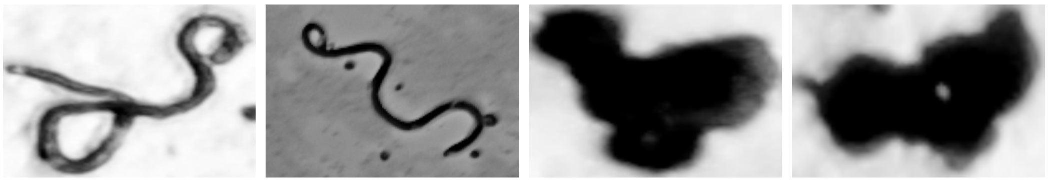
   where $|\nabla \boldsymbol{S}_{\hat{\boldsymbol{q}}}|$ is the gradient magnitude of $\boldsymbol{S}$ at $\hat{\boldsymbol{q}}$.

   An edge detector response (Figure 6.4b) can replace the gradient magnitude $|\nabla \boldsymbol{S}_{\hat{\boldsymbol{q}}}|$ in Equation 6.4. Since some points may have different radii, r is scale normalized to a predefined scale. In other words, $|\nabla \boldsymbol{S}_{\hat{\boldsymbol{q}}}|$ is calculated at $\hat{\boldsymbol{q}}$ within the original radius of $\boldsymbol{p}$, and then, $|\nabla \boldsymbol{S}_{\hat{\boldsymbol{q}}}|$ is mapped to a fixed-sized $\boldsymbol{D}$ via interpolation.

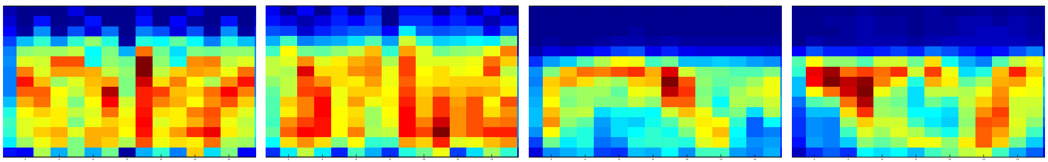
3.4.1 Dimensionality Reduction and Normalization

After calculating $\boldsymbol{D_p}$ for all points in the medial axis $\zeta(P)$, a single descriptor with a fixed size is required to fingerprint the object. One solution would be to learn a codebook as in [FFP05, NJT06], or to use other approaches like the Fisher vector [PD07]. Since the microfilaria is a featureless object, creating a codebook might not be effective due to the low number of "visual words" it has. The same problem might arise when using the Fisher vector approach.

In this work, the feature vector results from the vectorization of the sum of all descriptors for all points. To clarify, the resulting descriptor of an object P, is the HPC descriptor, *i.e.* a matrix of size $r \times k$, which is the sum of all $\boldsymbol{D_p}$ matrices, $\forall \boldsymbol{p} \in \zeta(P)$. The resulting matrix is vectorized and normalized by using the ℓ_1-norm. Examples of HPCs for different objects are shown in Figure 6.5. In this figure, despite the clear differences between the first and second images, they have quite similar HPCs. When microfilariae images are compared to non-microfilariae images the differences in HPCs are noticeable.



(a) Samples.



(b) HPCs.

Figure 6.5: HPCs calculated for different microfilaria and non-microfilaria objects. Top: the source images; Bottom: the corresponding HPC.

3.4.2 Incorporating Color

Color is a very important feature in objects. In light microscopy, intensity and color variations affect the objects and the background. Several reasons contribute to these variations. The model of the imaging devices may result in noticeable variation, color characteristics, filters, and concentration and acidity of the staining dye. Color variations are more difficult to handle than grayscale, since the intensity variations are encoded in the color space channels along with the color variations. However, color space transformations greatly help coping with many variations. Separation between intensity and color information may be of major importance. Color spaces that are based on human perception of color (*e.g.* HSV and CIELAB) are good for comparing color feature vectors. Device independence color spaces *e.g.* CIELAB, are in principle more advantageous than device dependent (*e.g.* RGB). In fact, it is not very clear in the literature what is the best color space to use for microfilariae. Therefore, experimental evaluation will help to asses which color space is more appropriate for this specific problem. Color information can be added to HPC in three main ways:

1. Concatenating the color features' vector with the HPC feature vector.
2. Calculating the HPC on all color channels and concatenating the HPCs' vectors.
3. Merge the results after separately classifying or matching the HPC and color feature vectors.

In the first approach, a color space, or a set of different color spaces' channels, can be used. A histogram is calculated and normalized for each channel [SB91] and, then, a color feature vector is formed for all channels by concatenating the channels' histograms. This vector is concatenated with the HPC feature vector to form a longer feature vector, which is normalized again. In the second approach, the HPC is calculated from all channels and the resulting vectors are concatenated. The last approach is different from the other two, where, instead of combining the feature vectors, the scores separately matching or classifying the HPC and color feature vectors are combined, *i.e.* two classifiers are used, one for the HPC and another one for the color features. The classifiers' scores are then ensembled to obtain a single score. In practice, the second approach did not perform well, because edge detection in different channels was not stable, therefore, only the first and third approaches will be reported in the results.

3.4.3 Multi-classifier Ensembling

Since the output of the decision function in Equation 4.16 are rigid values, *i.e.* a label $\{-1, 1\}$, it is difficult to establish an accurate multi-classifier ensembling strategy. However, literature [Kun04] is rich with many approaches to combine two (or more) homogeneous or heterogeneous classifiers. In this work, simple rules based on posterior probability estimation from the SVM output [Pla99] are used. The rule used in this work is to take the maximum between the posterior probabilities of the scores of the two classifiers.

3.4.4 Complexity

Although speed is not a major concern in the problem at hand, it is important to calculate the HPC's complexity for any future improvement. Let h be the number of pixels in an image, the complexity of detecting the hypothesis by the proposed approach can be calculated as follows. The top-hat transform requires $\mathcal{O}(h \times \alpha \times \beta)$, where α, β are the height and width of the kernel, respectively. For a fixed-size kernel, the complexity becomes $\mathcal{O}(h)$. Calculating mean-c using the integral image requires $\mathcal{O}(h)$.

In order to calculate HPC (6.2), both the connected component labeling (CCL) and boundaries' calculation require $\mathcal{O}(h)$. The Euclidean distance map (EDM) using Meijster's approach [MRH02] can be performed in linear time and requires $\mathcal{O}(h)$, and thinning requires $\mathcal{O}(h \log h)$ using [TVW02].

4 Experimental Results

Tests were designed to study how the proposed approach performs with and without color features. The proposed HPC is compared to two well-established academic approaches, SIFT and LIOP. Color histograms [SB91] on several color spaces: RGB, HSV, CIELAB, CIELUV, Opponent, XYZ, and CMYK, were also evaluated then, several combinations between HPC and color approaches were assessed. In order to further evaluate the proposed detection approach, it is compared to one of the most successful object detectors in the literature, part-based HOG [FGMR10]. The reason why this approach was chosen is three fold: first it is generic, second it has been recommended by leading researchers in object recognition, and third, it copes with objects deformations. The evaluation measures used to compare the descriptors are described in Table 6.1.

4.1 Materials

The materials used in this work are from three different sets of 26 images produced at the Institut de Recherche pour le Développement (IRD) Montpellier - France. These are:

Table 6.1: Measures used in the evaluation, where TP is the True Positives, TN is the True Negatives, FP is the False Positives, and FN is the False Negatives.

Measure	**Description**	**Calculation**
TPR	True Positive Rate (Recall)	$\frac{\mathrm{TP}}{\mathrm{TP+FN}}$
TNR	True Negative Rate (Specificity)	$\frac{\mathrm{TN}}{\mathrm{TN+FP}}$
PPV	Positive Predictive Value (Precision)	$\frac{\mathrm{TP}}{\mathrm{TP+FP}}$
NPV	Negative Predictive Value	$\frac{\mathrm{TN}}{\mathrm{TN+FN}}$
Accuracy		$\frac{TP+TN}{TP+FP+TN+FN}$

- Set A (Figure 6.6): Four 32-bit TIFF RGB images of size 2456×1632 pixels collected from unknown devices.
- Set B (Figure 6.7): Seven 32-bits JPG RGB images of size 3840×2160 pixels collected from unknown devices.
- Set C (Figure 6.8): Eleven 32-bit BMP RGB images of size 2048×1536 pixels obtained from unknown devices, nine of them were taken with $4\times$ magnification and, the other two, with $10\times$ magnification. This set also includes four 32-bit JPEG RGB images of size 4288×2848 pixels obtained from Zeiss Scope A1 Axio microscope with a Nikon D90 camera mounted on it, with ISO set to 400. Images in this set are manually color normalized [JRW97].

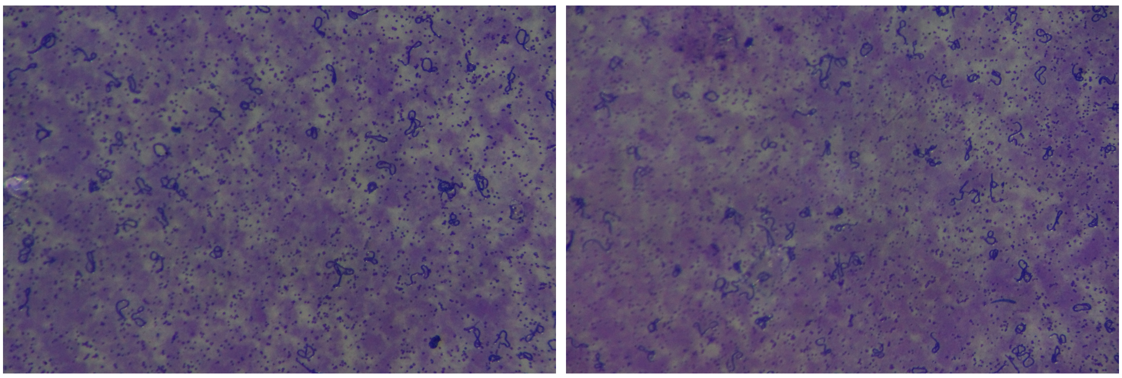


Figure 6.6: Set A.

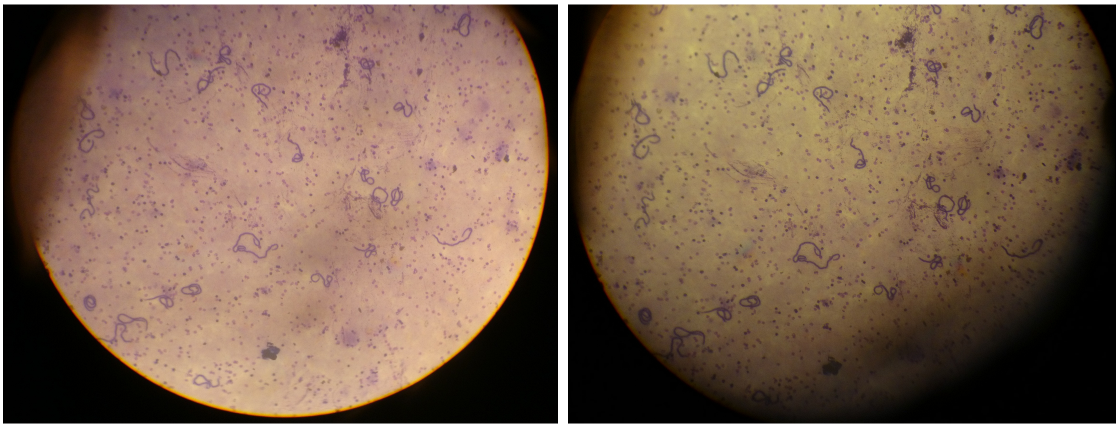


Figure 6.7: Set B.

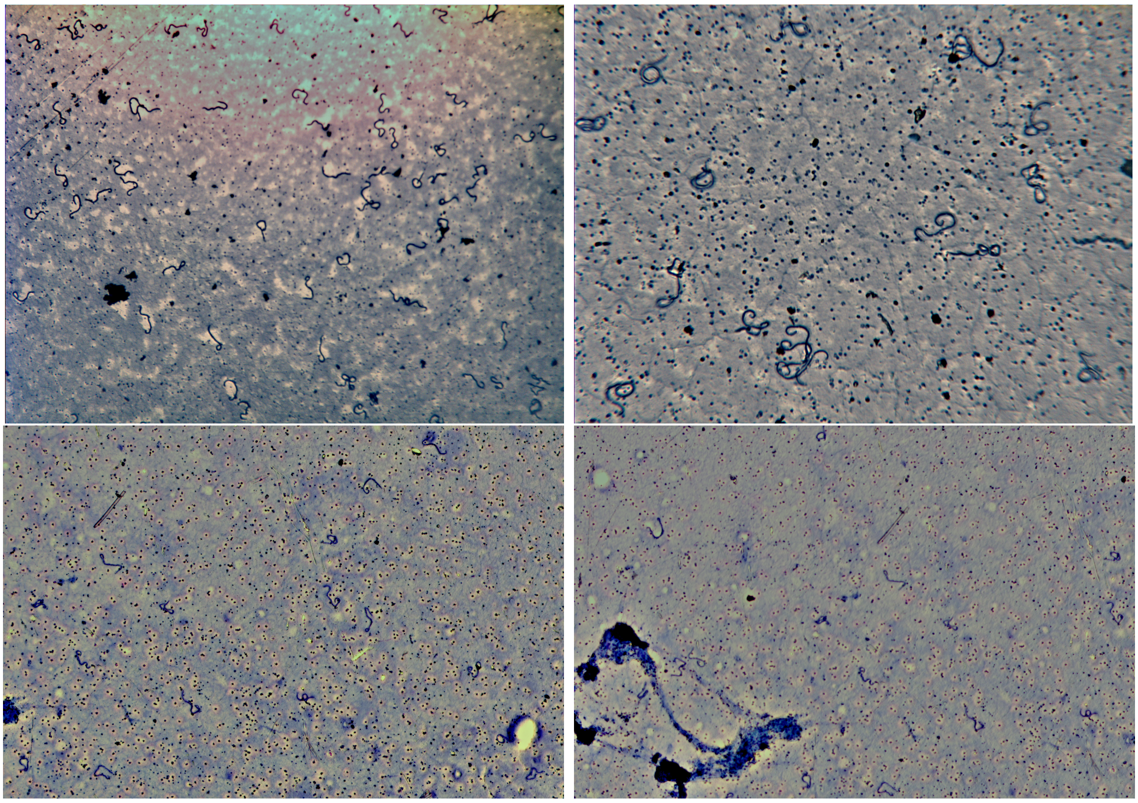


Figure 6.8: Set C, the top ones were taken from unknown devices, and the bottom ones were taken from a Zeiss Scope A1 Axio microscope.

4.2 Annotation

All images in the datasets are labeled by manually assigning a number and a label to each object: microfilaria, non-microfilaria, and undefined (Figure 6.9). A padding of 20 pixels was added to the bounding box for better centering the microfilariae in the sub-image.

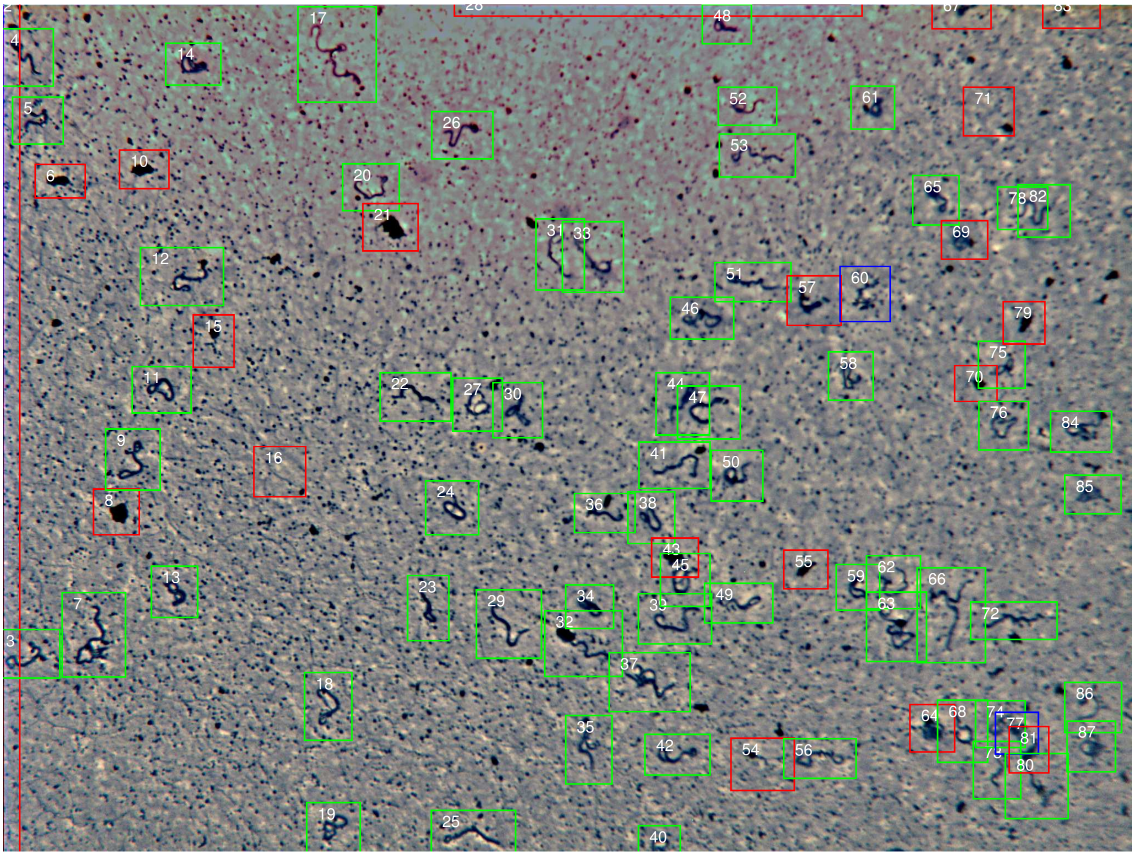


Figure 6.9: An annotated sample. Microfilaria, non-microfilaria, and undefined objects are labeled by green, red, and blue, respectively.

4.3 Parameters

For hypothesis generation, both width and height of the local window were set to 251 pixels and $c = 0.08$ (see Section 5.1.2 of Chapter 2). The minimum length of an object was set to 70 pixels. Any object shorter than this length was neglected from the detection process.

Since calculating SIFT on different images will produce a different number of key-points and, hence, feature vectors with different lengths, BoW of Dense-SIFT [VF10] was used. Each sub-image was sampled to non-overlapping regions of size 7×7 pixels, and then, SIFT was calculated on each region. This yields for each sub-image, a matrix with rows equal to the number of regions, and 128 columns (SIFT feature vector length). A codebook is created by randomly selecting 25% of the dataset, calculating Dense-SIFT, and then clustering to 300 classes (visual words). The clustering method was k-means using the Elkan's algorithm [Elk03]. This way, the input vector of SIFT will be of length 300. For more details see Appendix B. Likewise, for LIOP, a codebook of 300 classes was created. The number of neighbors was set to 4 and the radius to 5. Part-based HoG foreground overlap was set to zero to cope with the elasticity of the microfilaria. For color histograms each channel was pooled in 32-bins histograms. For HPC, the radius and number of orientations, were set to 16 and 15 respectively, and the input image was smoothed by a 3×3 Gaussian kernel with standard deviation of 1.

C-SVM was used for training. The kernel function was the radial basis function with radius of 10, and the Sequential Minimal Optimization (SMO) was the optimizer. The output of the classification was mapped to posterior probability using [Pla99].

4.4 Comparing the Descriptors

4.4.1 Generic Features Only

In this test, 10-fold cross validations were performed on the whole dataset (*i.e.* 2832 patches with 47% microfilaria and 53% non-microfilaria ratios) to evaluate HPC, SIFT, LIOP, and color histograms of RGB, HSV, CIELAB, CIELUV, Opponent, CMYK, and XYZ color spaces. It deserves noting that, HPC was calculated for the perimeter on the Maximum Connected Component (MCC) to reduce the background clutter.

The purpose of this "generic features" test was to evaluate existing descriptors without any features' combination, in other words, to evaluate only color or only grayscale features, but not combined. Table 6.2 depicts the average of all tests for binary classification measures.

Table 6.2: Averaged 10-fold cross-validations binary classification measures.

	Recall	Specificity	Precision	NPV	Accuracy
HPC	0.8789	0.9176	0.9003	0.9017	0.9008
SIFT	0.7873	0.9152	0.8881	0.8369	0.8573
LIOP	0.8043	0.9092	0.8826	0.8483	0.8623
RGB	0.7733	0.8395	0.8022	0.8162	0.8104
HSV	0.7725	0.8173	0.7806	0.8125	0.7984
LAB	0.4231	0.8237	0.6703	0.6306	0.6419
LUV	0.4203	0.8347	0.6827	0.6324	0.6469
Opponent	0.8045	0.8363	0.8047	0.8373	0.8224
XYZ	0.4189	0.8051	0.6419	0.6232	0.6299
CYMK	0.4447	0.8087	0.6626	0.6348	0.6437

In Table 6.2, HPC outperforms all methods in all measures. Opponent color space beats the other color spaces and, surprisingly, CIELAB and CIELUV, which are shown to be working well for similar images [LLLY12, CM02], produced poor results.

For a better understanding of the performance of each method, and to show the trade-off between recall and specificity, which is one of the main concerns of this test, the Receiver Operating Characteristic (ROC) curve is calculated for each method, and shown in Figure 6.10. This curve is more advantageous than other measures, since it normalizes the TPR and false positive rate (FPR, *i.e.* 1 - TNR) with respect to the number of positive and negative samples in the population.

In Figure 6.10, HPC shows the highest Area Under the Curve (AUC) among all other descriptors. At low FPR (< 0.1), HPC shows a distinctively better performance than the others. Although in Table 6.2 Opponent color space showed better recall

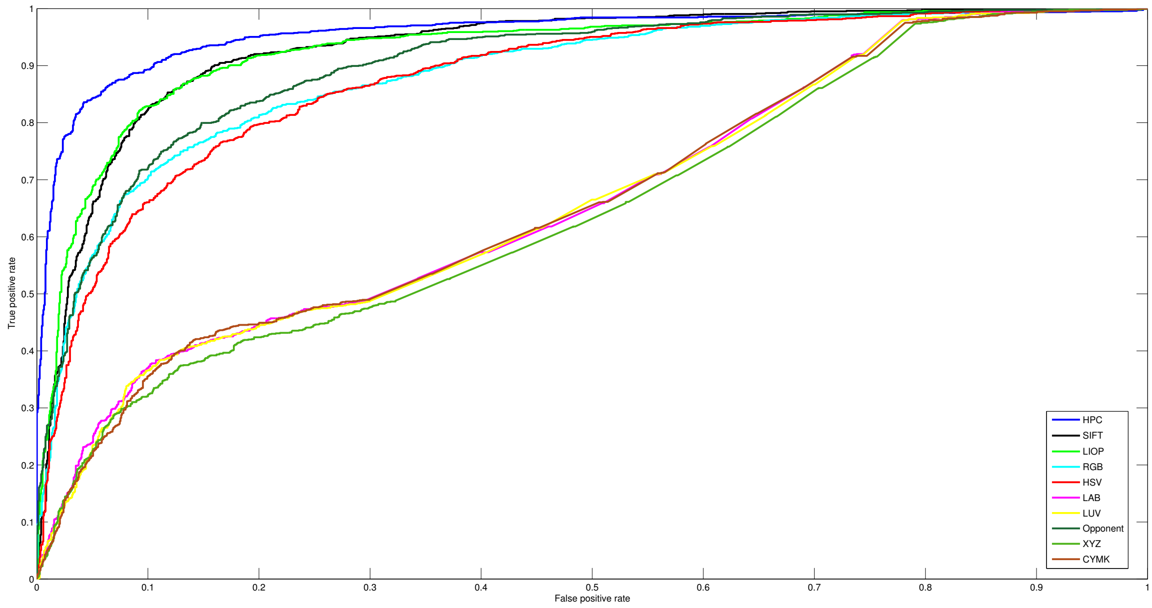


Figure 6.10: ROC Curve for the generic descriptors.

than SIFT and LIOP, it has a lower AUC. Other color spaces, except RGB and HSV, are performing very near to the random line.

In order to also assess the precision-recall measures, the Precision-Recall Curve (PR Curve) is shown in Figure 6.11. In this figure, HPC shows the best results again, and a very good trade-off between recall and precision. SIFT and LIOP show very similar results, and Opponent's results are the best among all color spaces.

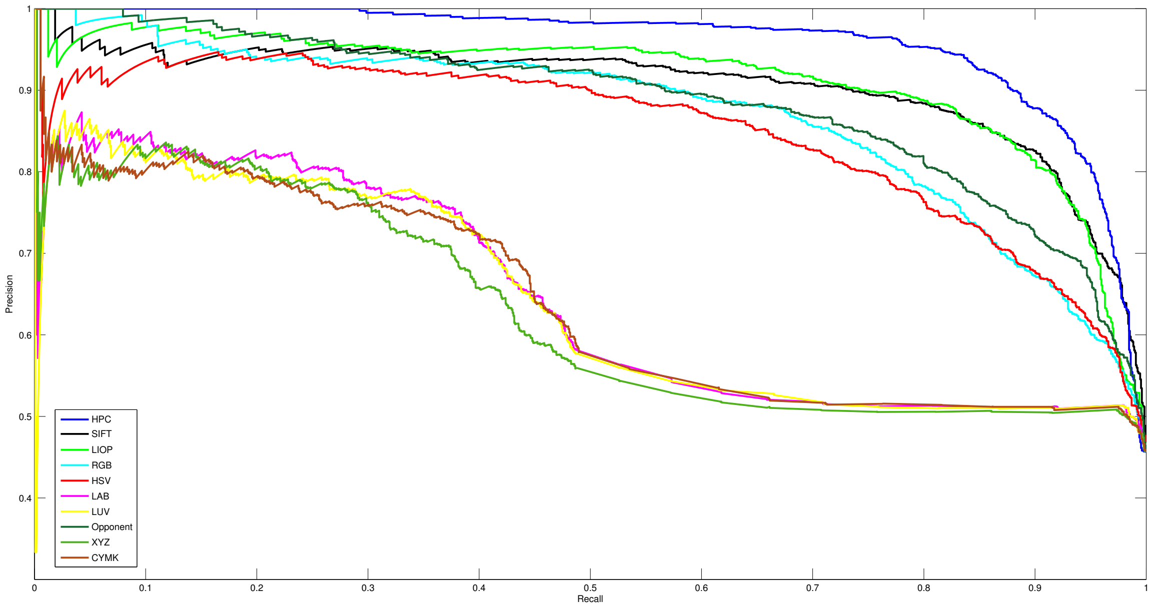


Figure 6.11: PR Curve for the generic descriptors.

4.4.2 Incorporating Color

Since the Opponent color space has shown to produce better results than other color spaces, and, in addition, it has been shown to be more robust to light and color variation [vdSGS10], it will be incorporated with HPC, SIFT and LIOP in the current test. This color space will be incorporated with these descriptors as described in Section 3.4.2. In addition, the Opponent-SIFT method described in [vdSGS10] will also be evaluated in this test.

The averaged 10-fold cross validations are depicted in Table 6.3, and the corresponding ROC and PR Curves are shown in Figures 6.12, and 6.13, respectively.

Table 6.3: Averaged measures for 10-fold cross validations for combined features.

	Recall	Specificity	Precision	NPV	Accuracy
Concat. of HPC and Opponent	0.8759	0.9374	0.9237	0.9021	0.9107
Max of HPC and Opponent	0.9204	0.9415	0.9311	0.9357	0.9329
Concat. of SIFT and Opponent	0.7604	0.9177	0.8865	0.8210	0.8467
Max of SIFT and Opponent	0.8588	0.9104	0.8902	0.8857	0.8877
Concat. of LIOP and Opponent	0.7696	0.9128	0.8836	0.8270	0.8489
Max of LIOP and Opponent	0.8635	0.9053	0.8852	0.8893	0.8870
Opponent-SIFT [vdSGS10]	0.8894	0.9330	0.9191	0.9089	0.9135

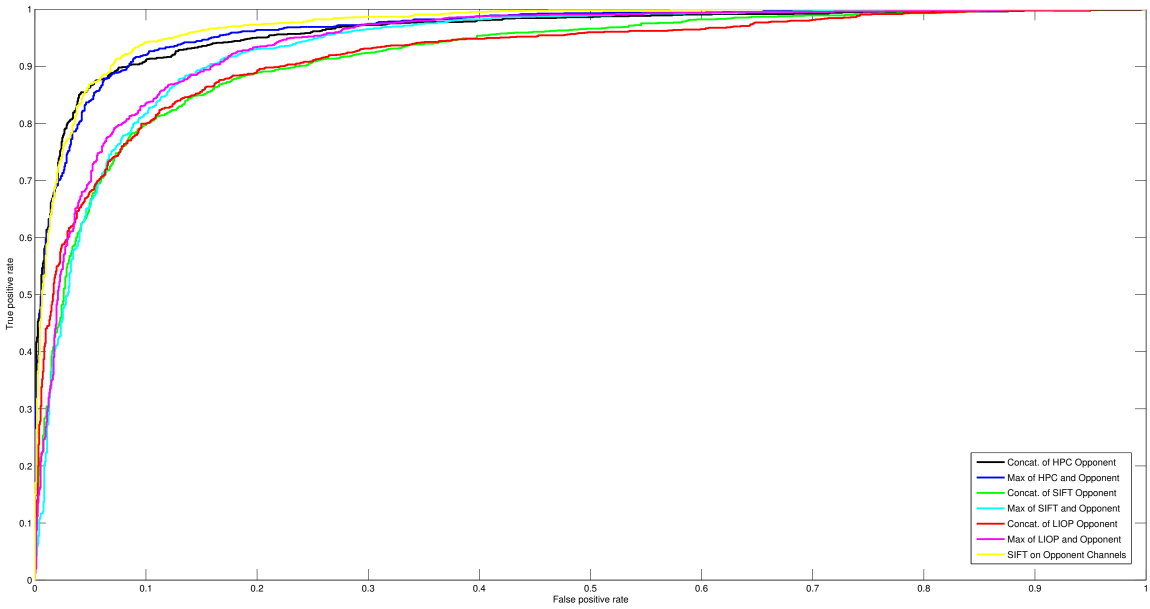


Figure 6.12: ROC Curve for the combining shape and color features.

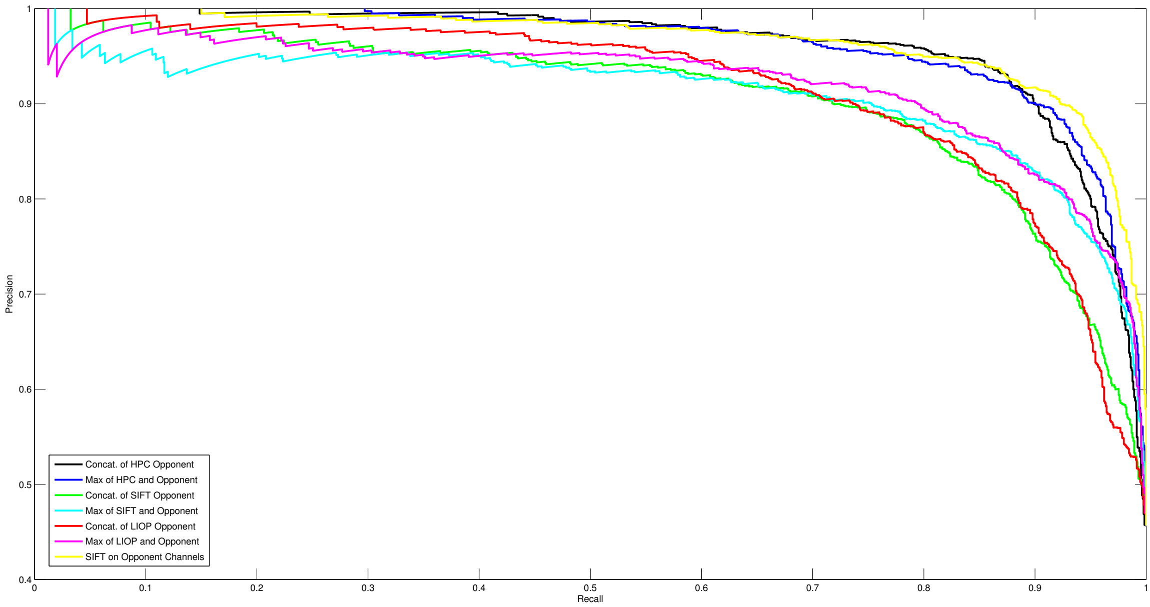


Figure 6.13: PR Curve for the shape and color descriptors.

Although in Table 6.3, the score resulting from merging Opponent and HPC produced the best results in all measures, Opponent-SIFT is slightly better in Figures 6.12 and 6.13. This slight difference can be seen in the AUC measures, where the Max of HPC and Opponent produced 0.9660 and Opponent-SIFT produced 0.9725. Considering other approaches, and comparing Tables 6.2 and 6.3, it can be noticed that, merging the scores between color and shape produces better results than concatenating the feature vectors.

4.4.3 Set Testing

In the previous tests, the whole set of patches was used, where the training and testing subsets in each fold contained images from all sets. Therefore, the color variation between the sets was not validated. In a real case scenario, it is common to test images with different colors being captured from different imaging devices, other than the ones used to obtain the training set.

In this test, two sets will be used for training, and a set will be used for testing. The objective is to show the ability the previously tested descriptors have in coping with appearance variation. Table 6.4 depicts the results of the tested methods, where sets A and B were used for training and set C was used for testing. The ROC and PR Curves are shown in Figures 6.14 and 6.15, respectively. The same is shown for testing with set B and set A, in Tables 6.5 and 6.6, as well as the respective ROC Figures 6.16 and 6.18, and PR Figures 6.17 and 6.19.

Table 6.4: Testing results for training with sets A and B and testing set C.

	Recall	Specificity	Precision	NPV	Accuracy
HPC	0.8496	0.9330	0.8970	0.9003	0.8990
SIFT	0.9024	0.9134	0.8775	0.9316	0.9089
LIOP	0.3821	0.9860	0.9495	0.6990	0.7401
Opponent	0.7236	0.8073	0.7206	0.8095	0.7732
Concat. of HPC and Opponent	0.7236	0.9162	0.8558	0.8283	0.8377
Max of HPC and Opponent	0.8780	0.9469	0.9191	0.9187	0.9189
Concat. of SIFT and Opponent	0.6545	0.8966	0.8131	0.7906	0.7980
Max of SIFT and Opponent	0.9106	0.9218	0.8889	0.9375	0.9172
Concat. of LIOP and Opponent	0.2439	0.9637	0.8219	0.6497	0.6705
Max of LIOP and Opponent	0.5366	0.9721	0.9296	0.7532	0.7947
SIFT on Opponent Channels	0.2114	0.9972	0.9811	0.6479	0.6772

Table 6.5: Testing results for training with sets A and C and testing set B.

	Recall	Specificity	Precision	NPV	Accuracy
HPC	0.7725	0.9307	0.9211	0.7962	0.8498
SIFT	0.3360	0.9584	0.8944	0.5796	0.6401
LIOP	0.6138	0.9584	0.9393	0.7033	0.7821
Opponent	0.2063	0.9751	0.8966	0.5399	0.5819
Concat. of HPC and Opponent	0.4921	0.9584	0.9254	0.6431	0.7199
Max of HPC and Opponent	0.5291	0.9723	0.9524	0.6635	0.7456
Concat. of SIFT and Opponent	0.2566	0.9695	0.8981	0.5547	0.6049
Max of SIFT and Opponent	0.2513	0.9723	0.9048	0.5536	0.6035
Concat. of LIOP and Opponent	0.3810	0.9751	0.9412	0.6007	0.6712
Max of LIOP and Opponent	0.4630	0.9723	0.9459	0.6336	0.7118
SIFT on Opponent Channels	0.3757	0.9917	0.9793	0.6027	0.6766

Table 6.6: Testing results for training with sets B and C and testing set A.

	Recall	Specificity	Precision	NPV	Accuracy
HPC	0.8728	0.8988	0.8754	0.8966	0.8871
SIFT	0.5374	0.9012	0.8159	0.7052	0.7379
LIOP	0.5195	0.9195	0.8402	0.7014	0.7399
Opponent	0.2829	0.8463	0.6000	0.5916	0.5934
Concat. of HPC and Opponent	0.6407	0.9634	0.9345	0.7670	0.8185
Max of HPC and Opponent	0.7710	0.9195	0.8864	0.8313	0.8528
Concat. of SIFT and Opponent	0.3383	0.9756	0.9187	0.6441	0.6895
Max of SIFT and Opponent	0.4731	0.9183	0.8251	0.6814	0.7184
Concat. of LIOP and Opponent	0.3728	0.9573	0.8768	0.6520	0.6949
Max of LIOP and Opponent	0.4506	0.8976	0.7818	0.6673	0.6969
SIFT on Opponent Channels	0.0120	0.9988	0.8889	0.5538	0.5558

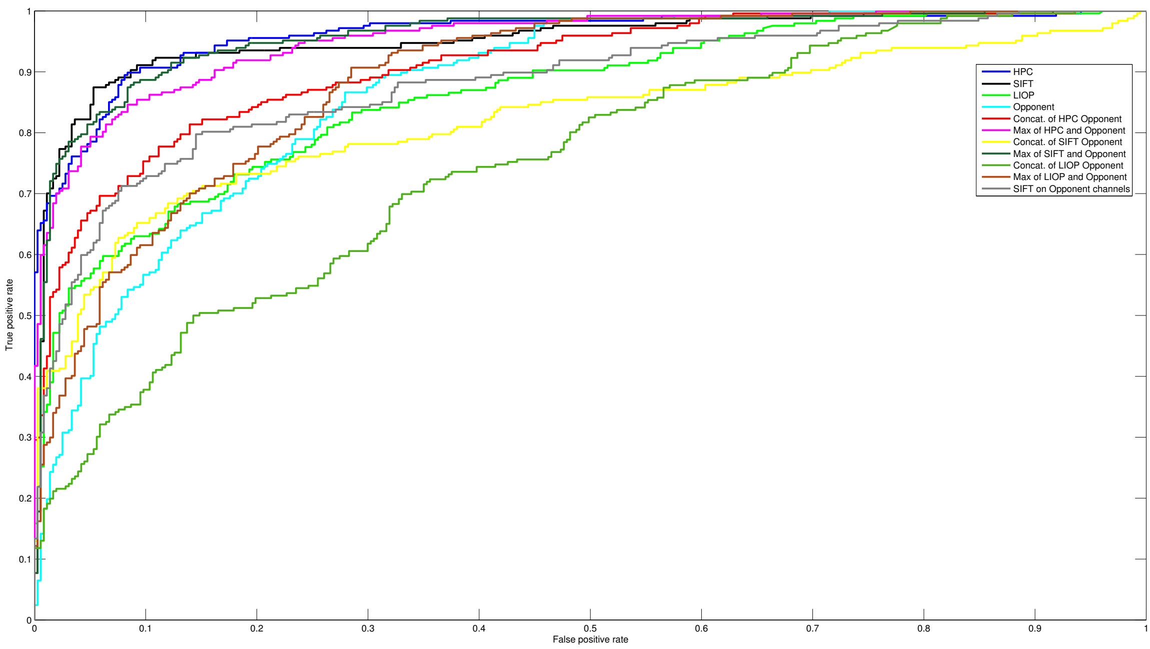


Figure 6.14: ROC Curve for training using sets A and B and testing set C.

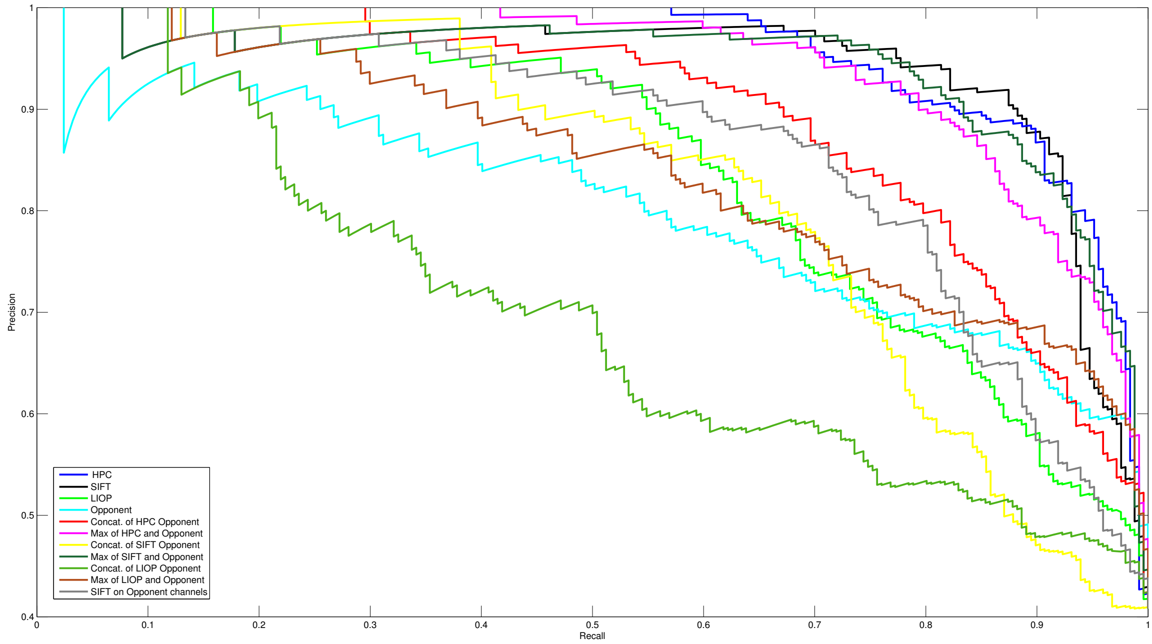


Figure 6.15: PR Curve for training using set A and B and testing set C.

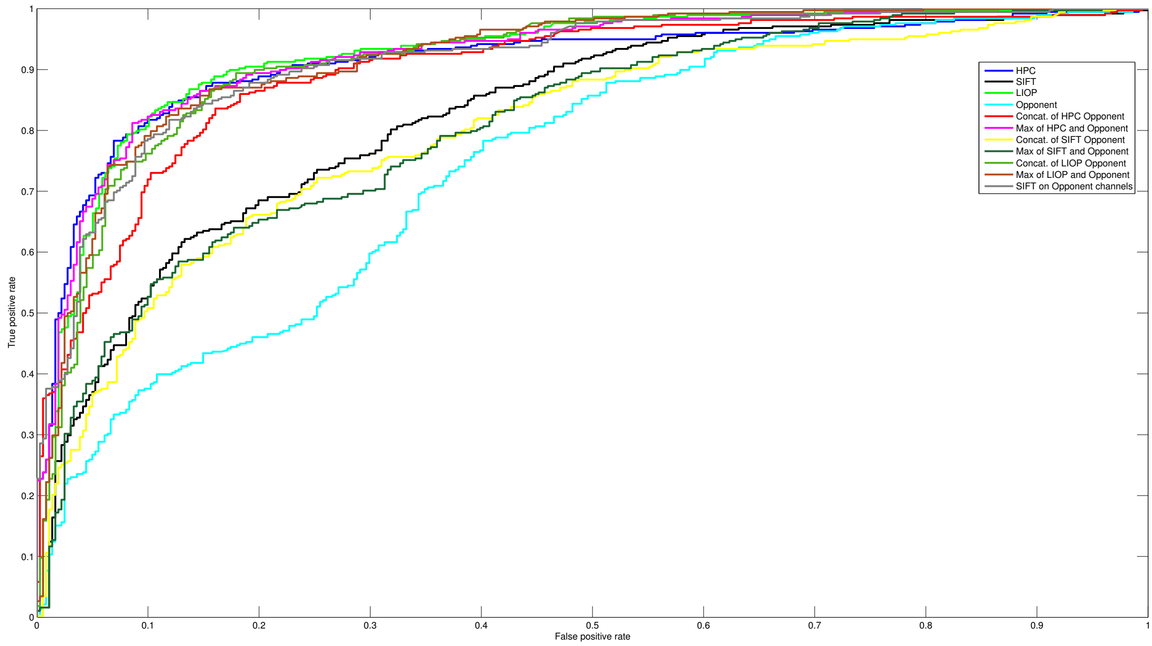


Figure 6.16: ROC Curve for training using set A and C and testing set B.

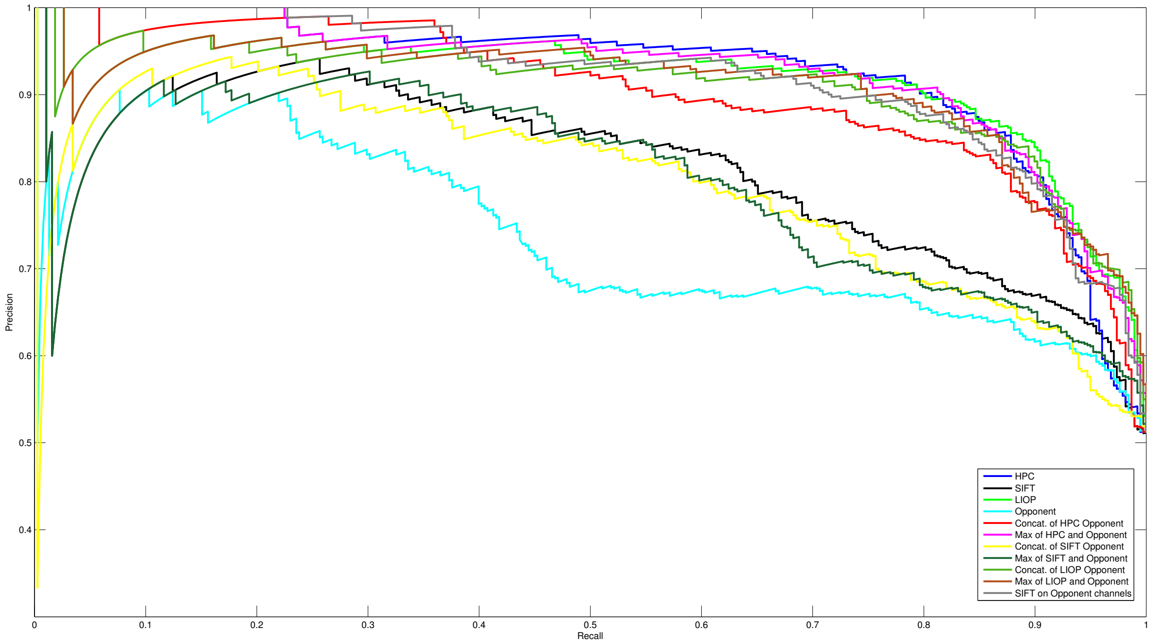


Figure 6.17: PR Curve for training using set A and C and testing set B.

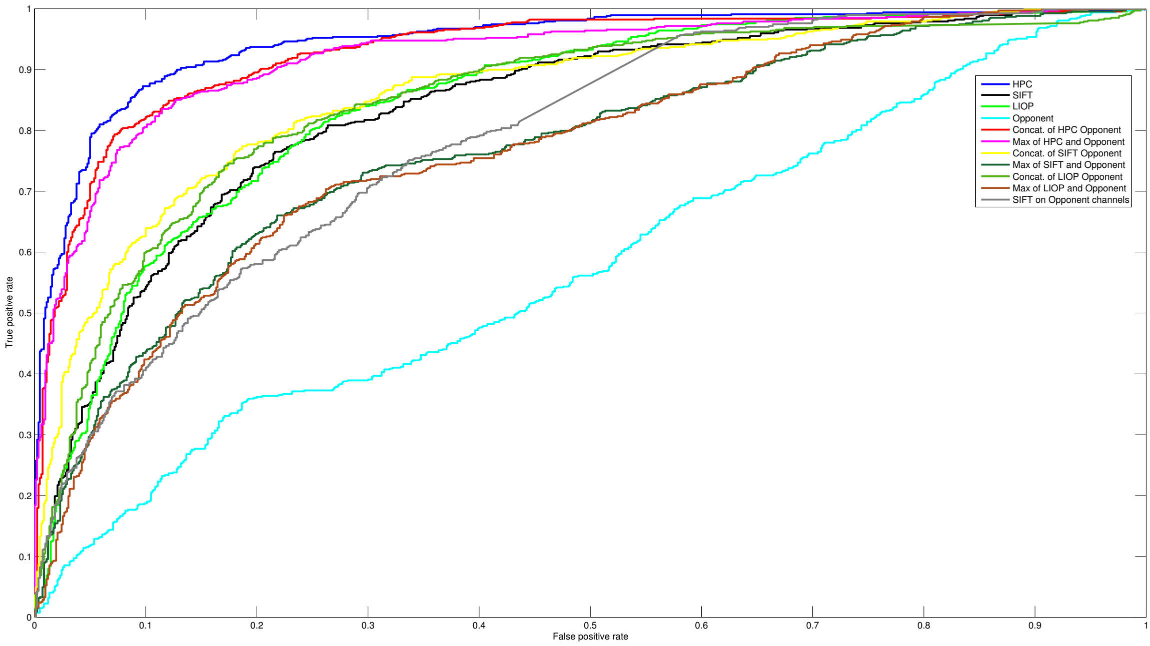


Figure 6.18: ROC Curve for training using set B and C and testing set A.

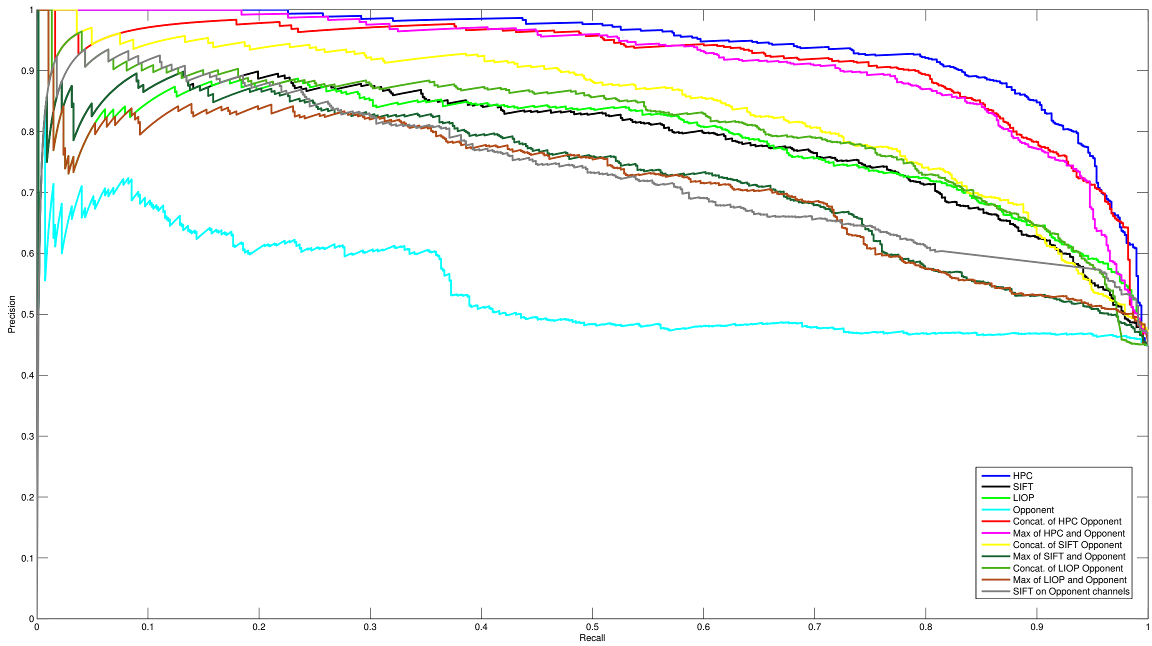


Figure 6.19: PR Curve for training using set B and C and testing set A.

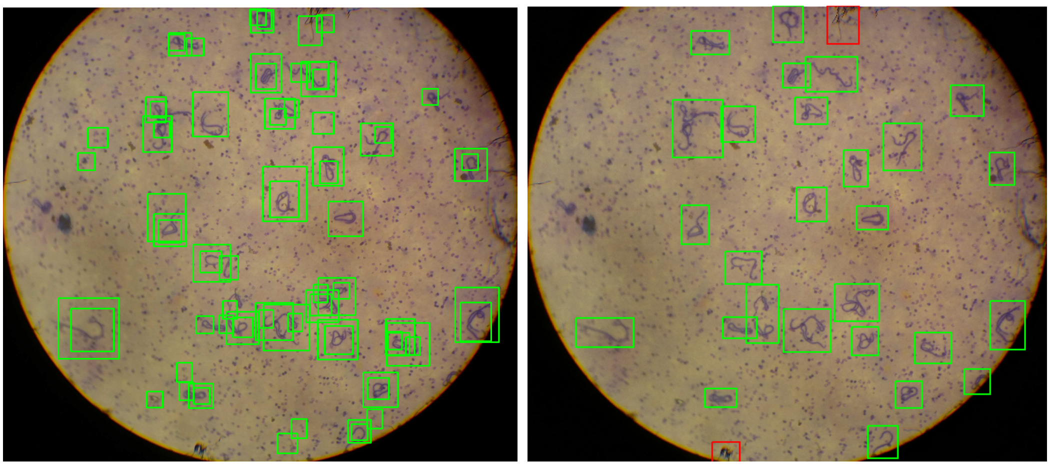
Tables 6.4, 6.5, and 6.6 show the superiority of HPC when compared to all other methods, including the concatenation between HPC and Opponent color space. Color spaces were producing poor results, and are not shown in the Tables, except for the opponent color space. HPC showed to be very stable among all sets. On the other hand, methods that showed impressive results when testing the whole set (Table 6.3 and Figures 6.12 and 6.13), performed poorly in the current test. For instance, Opponent-SIFT showed very poor results when compared to SIFT in Table 6.4.

4.5 Comparison with "the" State-of-the-art Detector

Although the conducted tests have showed the ability of HPC to perform nicely in different cases, it is very important to test the whole detection system, by comparing it to the the part-based HOG, currently known as the best detector.

In this test, 9 images of 4× zoom were selected from sets B and C. This yields, about 1039 sub-images for training of which 477 (45.9%) are microfilaria and 562 (54.1%) are of non-microfilaria. The remaining 17 images were used for testing.

Since, in many cases, part-based HOG produces many detections for the same microfilaria, *i.e.* excessive partial detection, the proposed hypotheses generator was used to generate hypotheses for part-based HOG. Figure 6.20 shows the detection results of part-based HOG with its own hypothesis generator and with the proposed one.



(a) Part-based HOG detection using its own hypothesis generator.

(b) Part-based HOG detection using the proposed hypothesis generator.

Figure 6.20: Microfilariae detection using part-based HOG. Green and red boxes identify true positives and false positives, respectively.

The results in Table 6.7, regarding HPC, were obtained using a square window with size of 150 pixels, and $c = 0.08$. The minimum length was set to 50 pixels, and each hypothesis window was segmented using local mean with a small window of size 75. The HPC was calculated on the contour of the binary objects.

In Table 6.7, the number of detected microfilariae is estimated using the median

Table 6.7: Detection results. HPC on the left and part-based HOG on the right.

		HPC					Part-Based HOG				
#	GT	TP	FP	FN	Rec.	Pre.	TP	FP	FN	Rec.	Pre.
1	15	15	4	0	1.0000	0.7895	13	4	2	0.8667	0.7647
2	15	15	3	0	1.0000	0.8333	7	7	8	0.4667	0.5000
3	12	12	6	0	1.0000	0.6667	6	2	6	0.5000	0.7500
4	28	28	2	0	1.0000	0.9333	26	3	2	0.9286	0.8966
5	28	27	2	0	1.0000	0.9310	25	4	3	0.8929	0.8621
6	25	20	3	6	0.7692	0.8696	25	2	0	1.0000	0.9259
7	30	29	1	1	0.9667	0.9667	30	4	0	1.0000	0.8824
8	31	23	0	6	0.7931	1.0000	30	3	2	0.9375	0.9091
9	127	104	7	23	0.8189	0.9369	50	5	90	0.3571	0.9091
10	128	98	9	30	0.7656	0.9159	55	3	71	0.4365	0.9483
11	16	16	3	0	1.0000	0.8421	12	1	4	0.7500	0.9231
12	45	41	2	4	0.9111	0.9535	42	7	3	0.9333	0.8571
13	56	52	5	4	0.9286	0.9123	52	3	4	0.9286	0.9455
14	84	70	2	14	0.8333	0.9722	76	2	8	0.9048	0.9744
15	71	58	2	11	0.8406	0.9667	67	1	4	0.9437	0.9853
16	132	116	4	16	0.8788	0.9667	99	3	33	0.7500	0.9706
17	16	15	3	1	0.9375	0.8333	16	2	0	1.0000	0.8889
	859	**739**	**58**	**116**	**0.8643**	**0.9272**	**631**	**56**	**240**	**0.7245**	**0.9185**
Standard Deviation					**0.0892**	**0.0848**				**0.2192**	**0.1168**

length of the detected TP, since the median is resistant to outliers. In terms of recall, HPC outperforms the part-based HOG detector. For precision, they both produce very similar results. However, HPC is susceptible to detect thin objects, even though they are not microfilariae, and was not able to detect intensely overlapped microfilariae. Part-based HOG also detects objects which are not microfilariae, but sometimes it rejects clear microfilariae. The detection results of some samples are shown in Figure 6.21 and examples of these false negatives and positives for HPC and part-based HOG are shown in Figures 6.22 and 6.23, respectively.

In Figure 6.22, it is clear that, HPC misses some microfilariae which are even difficult to be detected by visual inspection. In addition, most of the false positives, are thin objects, which can also be difficult to distinguish visually. On the other hand, part-based HOG detects false positives, which are not even thin objects and, in some cases, it rejects true positives, which are distinctively microfilariae.

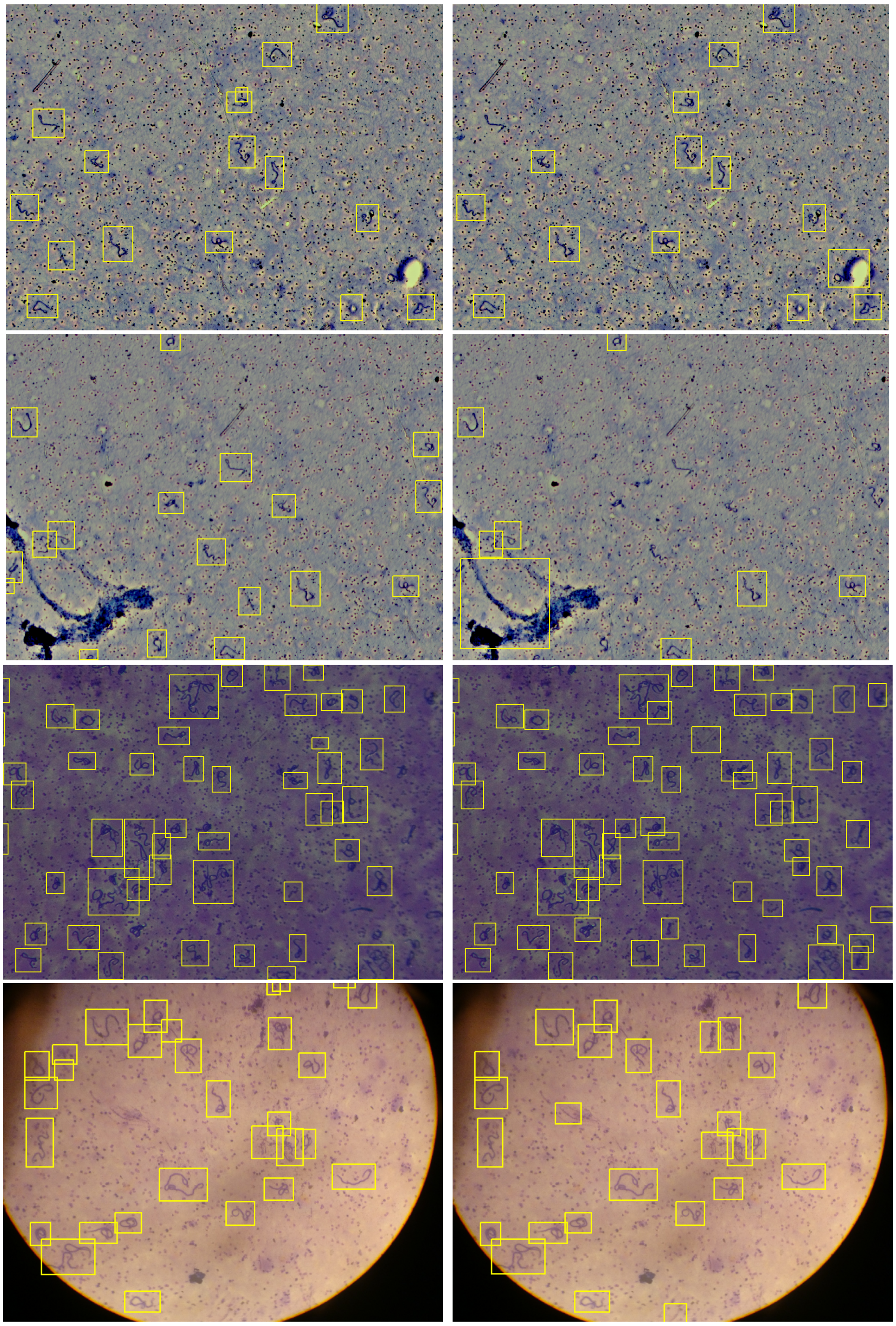
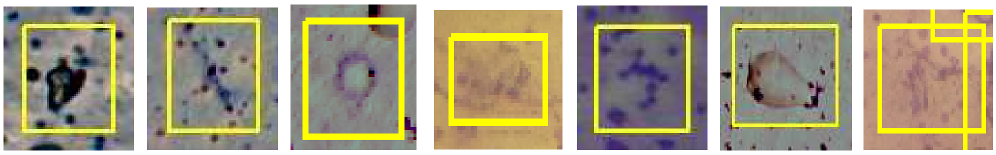


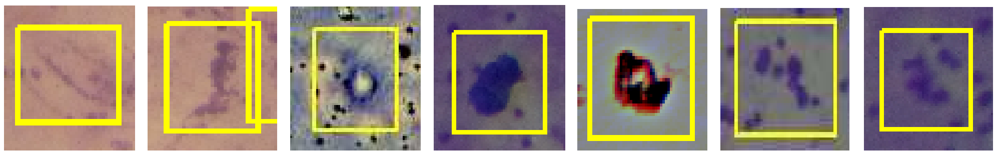
Figure 6.21: Detection results. Left: HPC; Right: part-based HOG.



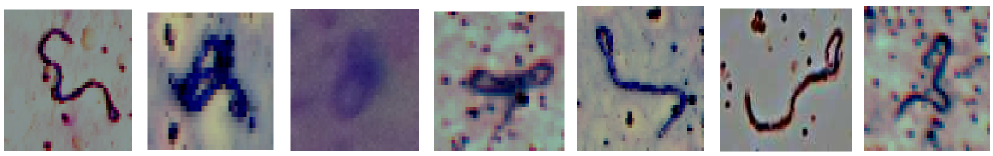
(a) False positives.

(b) False negatives.

Figure 6.22: False positive and false negative objects processed by HPC.



(a) False positives.



(b) False negatives.

Figure 6.23: False positive and false negative objects processed by part-based HOG.

5 Conclusions

Although microfilaria are just curvilinear objects, they overlap and deform in almost a free-form way. For this reason, descriptors that handle this variation are good choices. Global descriptors, like color histograms, are good when the color does not noticeably vary between the training and testing samples. Color histograms are invariant to some transformations *e.g.* rotation and, when normalized, they are robust to scaling, but they are sensitive to color change of the staining dye.

Keypoint-based, and patch-based descriptors also seem to be affected by variation between the sets. In other words, although the set test was designed to evaluate color features, it also showed that, these descriptors report unstable results when different sets are used for training and testing. On the other hand, the proposed approach, HPC, which can be regarded as a voting descriptor, where local points vote for a specific local structure, show stable results in all tests and, even if in one of the tests is slightly lower than the Opponent-SIFT, it reports the most stable results. In addition, when HPC detection is compared to one of the best state-of-the-art detectors, part-based HOG, HPC shows higher recall, but it may reject microfilariae when they overlap. Part-based HOG also rejects overlapped microfilariae. Moreover, part-based HOG, rejects many more obvious microfilariae and accepts many false microfilariae, when compared to HPC. Therefore, HPC is clearly the most robust and stable detector out of all the tested approaches.

7

APPLICATIONS

1 Introduction

DURING the work of this thesis, several types of images were studied. Several methods and applications were developed and are published, while some others are under preparation. This chapter presents some of them. In Section 2, a rice panicle analysis application is presented. A method proposed for rings artifacts suppression in X-ray images is presented in Section 3. In Section 4, an application to help users of microfilariae detection system to analyze and post-process the detection results easily, is presented.

2 Panicle and Grain Analysis

2.1 Background

In crops, inflorescence complexity and the shape and size of seeds are among the most important characteristics that influence yield. For example, rice panicles vary considerably in the number and order of branches, elongation of the axis, and the shape and size of the seed. Manual low-throughput phenotyping methods are time consuming, and the results are unreliable. However, high-throughput image analysis of the qualitative and quantitative traits of rice panicles is essential for understanding the diversity of the panicle as well as for breeding programs.

2.2 P-TRAP: a Panicle Trait Phenotyping Tool

This application (Figure 7.1) performs panicle and grain analysis [ATAA+13]. The input image is a 2-D panicle image, where the objective is to calculate the panicle traits, such as the number of primary, secondary, and tertiary branches and their length. In addition, the number and traits of grains on the panicles are calculated. The panicle structure is analyzed by means of image thinning and graph theory. The input image is segmented and the skeleton of the panicle is converted into a graph. Panicle traits can be extracted by the analysis of this graph. The grains are detected by mathematical morphology and shape analysis [DBB02]. A very friendly user interface is provided to allow users to perform the post processing efficiently. For instance, users can adjust the length of a branch, add, or delete a branch, and the application will adjust the panicle structure with the user corrections if needed.

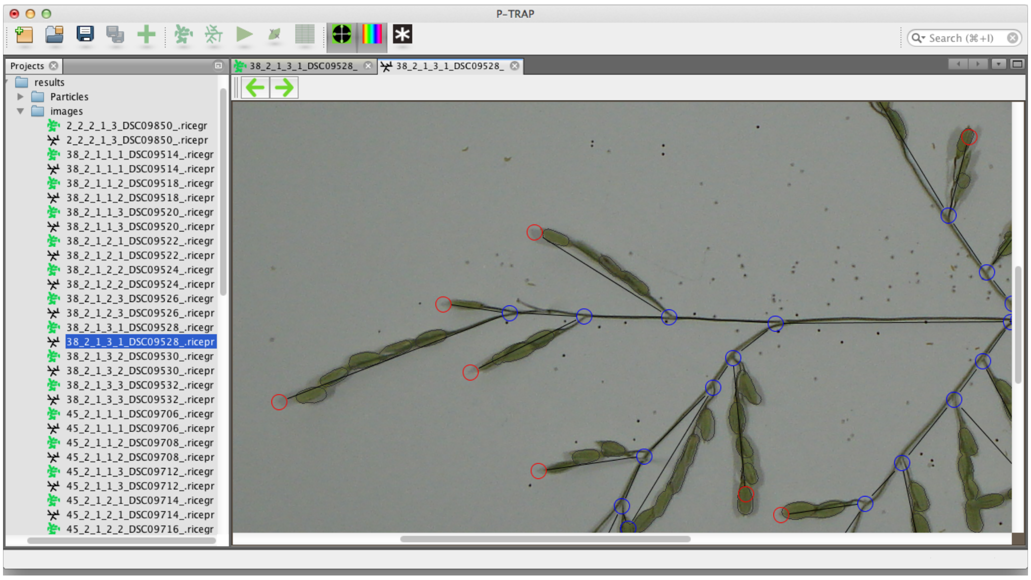
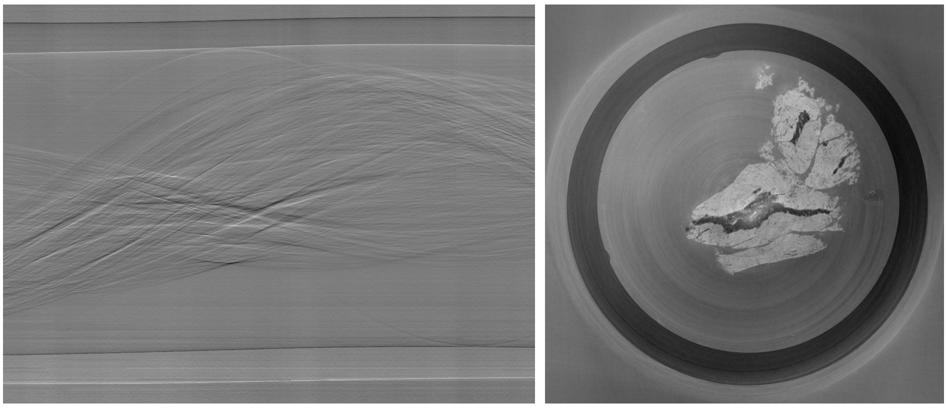


Figure 7.1: Panicle traits application.

3 X-ray Rings Artifact Suppression

X-ray images are widely used in many applications in medicine, geology, and food engineering. When X-ray interact with the matter, three main processes can be observed: Absorption, refraction, and scattering. In addition, there are three different modalities (images) resulting from these interactions: Transmission, differential phase-contrast, and dark-field. Differential phase-contrast is the preferred modality in food engineering because it reveals the soft tissues such as those found in the meat. For 3-D Computed Tomography (CT), the sample is managed by a rotation system to calculate the 3-D volume. There are several types of rotation systems, such as fan and parallel.

Grating-Based Imaging (GBI) systems are used to create a phase-contrast imaging. A GBI consists of two to three grates to generate a grating pattern, known as Talbot pattern. The variation of this pattern due to annotating the X-rays by a sample will produce changes. These changes (measurements) are recorded at the photo-detector behind the last grate of the GBI. To construct a slice, these measurements



(a) A sinogram. x and y-axis represent the detector samples r and angular variable θ, respectively.

(b) The reconstructed slice of a meat sample.

Figure 7.2: A phase-contrast slice calculated from its sinogram.

are back-projected in a process called filtered back-projection.

Each slice (Figure 7.2b) in the volume is constructed from the line integrals along all projections (the sinogram). The sinogram (Figure 7.2a) is a 2-D polar function, where one axis represents the detector samples r and, the other, represents the angular sampling θ [Hsi09]. It is usually modeled using the 2-D Radon transform:

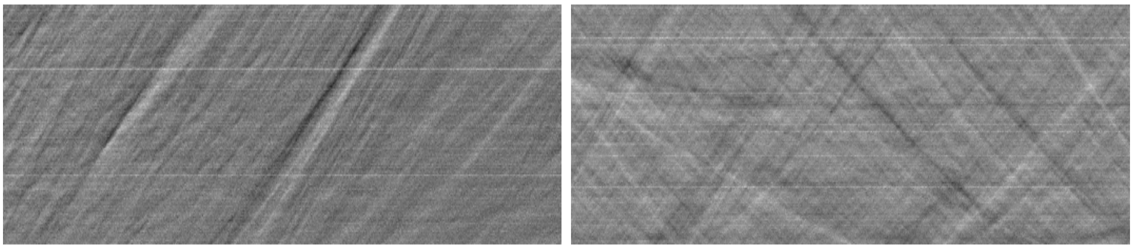
$$\mathcal{S}(r,\theta) = \int\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - r) \tag{7.1}$$

where $f(x,y)$ is the slice and δ is the Dirac delta function.

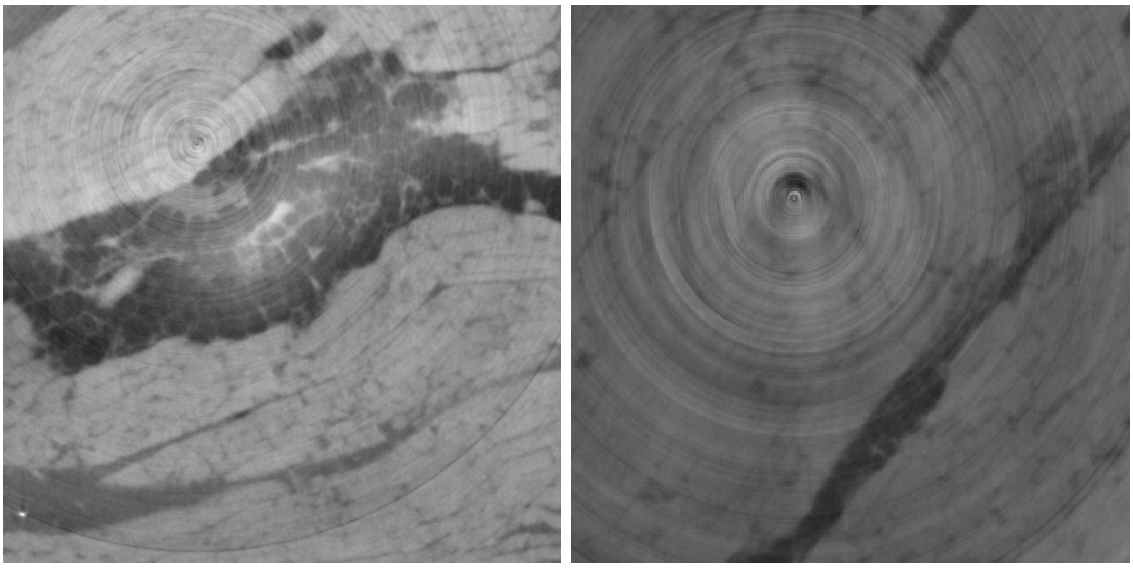
In order to construct $f(x,y)$ from $\mathcal{S}(r,\theta)$, the inverse Radon transform is used. This is equivalent to the Fourier slice theorem. By taking the Fourier transform of each measurement at each angle, and filling the 2-D Fourier space with respect to the angle, one can construct the slice by taking the inverse Fourier transform. Another way to view the reconstruction task is by using the stackgram [HA02].

3.1 Rings Artifacts

Defective and/or unevenly calibrated detectors do not respond linearly. Therefore some pixels in the reconstructed slices become erroneous. In the sinogram perspective, those pixels are modeled as lines (or line segments). In the corresponding slice, they are mapped to concentric circles or parts thereof. Processing slices affected by rings without correcting them is an extremely difficult task [JATD15].



(a) Sinogram bands with strips.

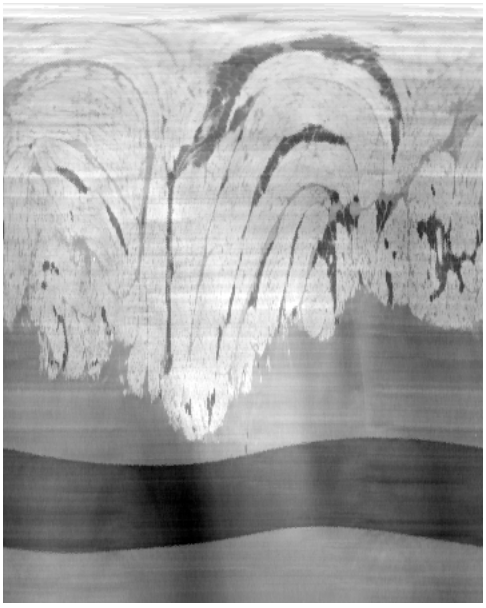


(b) Parts of the slices with rings artifacts.

Figure 7.3: Rings artifacts.

In order to remove rings, they have to be located beforehand. It is difficult to detect them because, in many cases they are incomplete, vary in intensity, and are not equidistant. In short, they do not follow a pattern. As the objective is to remove them without distorting the original structure in the slice, it is important to distinguish ring pixels from image structure pixels. Dealing with rings directly in the slice is cumbersome. It makes it easier to transform the circular segments into line segments. There are two ways to do that: pre and post reconstruction. In the former, sinograms are used (Figure 7.2) and in the latter, slices are transformed from the Cartesian domain to the polar domain (Figure 7.4). In the Polar domain the angular θ and radial r are represented in two axes. In both cases, the objective then becomes to locate lines that correspond to rings and to erode them.

(a) A slice.



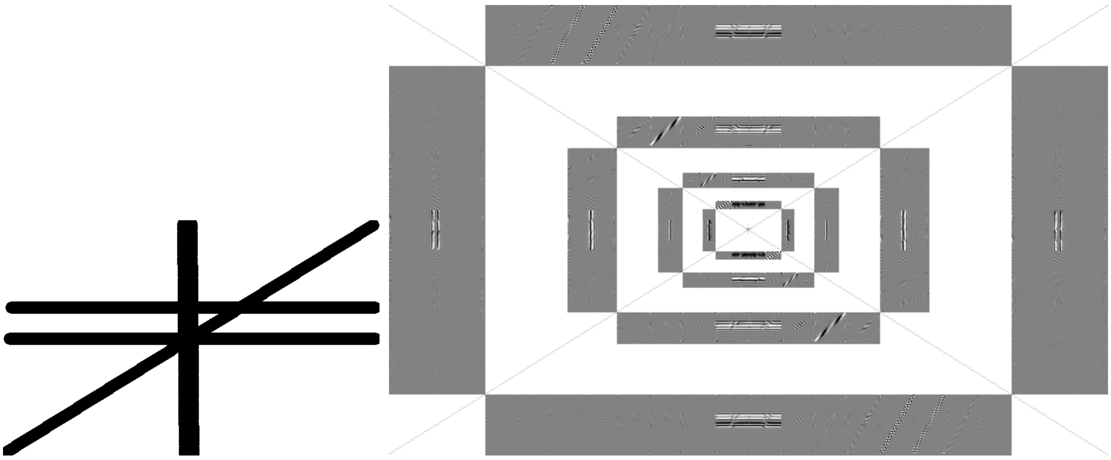
(b) The polar transform of the slice.

Figure 7.4: A phase-contrast slice transformed from the Cartesian to the polar domain.

3.2 Scale-dependent Suppression

When the curvelet transform is applied (Section 4.3.2 of Chapter 2) to a sinogram, the structures in the sinogram will be decomposed according to scale, orientation, and position. Figure 7.5 demonstrates the responses of the curvelet transform to linear structures of different positions, orientations and scales in a synthetic image.

In this figure, there are 4 quadrants centered around the low frequency component (ordered clockwise where the 1st quadrant is at the top), 5 scales, and 10 orientations. At the center lie the low frequency components. At each corona, there are different orientations, which map the responses to structures of the same scale but with different orientations.



(a) A sample synthetic image. (b) The curvelet transform tiling.

Figure 7.5: Discrete curvelet transform of a sample image.

There are several considerations to be made when designing the suppression method:

- The coefficients in the curvelet frame are perpendicular to the structures in images.
- The response of the lines increase when moving from the low frequency component to high frequencies (higher coronas).
- Lines in the central rows of the sinogram are mapped to smaller rings, while lines near to the borders of the sinogram are mapped to larger rings.

Concerning the first point, the horizontal lines will have high responses at the 1st and 3rd quadrants, and this guarantees the detection of the lines. For the second point, in order to remove the lines, the corresponding coefficients must be treated with respect to the scale, *i.e.* the level of the corona, so that the lines can be removed. Regarding the last consideration, lines must be treated according to their distance from the center of the sinogram. Therefore, it is important to correct the sinogram center r_{center}.

Once the lines have been detected in the 1st and 3rd coronas. The coefficient scale-dependent suppression function can be defined as:

$$c = \begin{cases} 0, & \text{if} \quad |c| < \frac{\psi(r;r_{\text{center}})}{j} \\ c, & \text{otherwise} \end{cases} \tag{7.2}$$

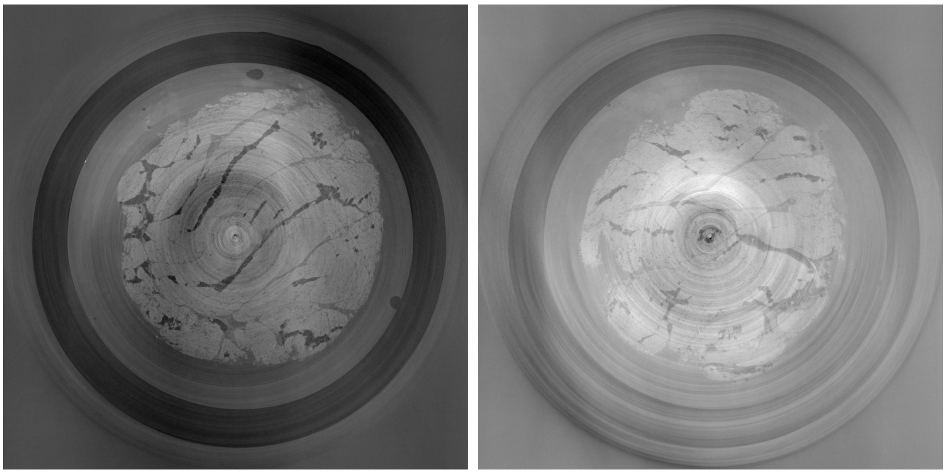
where c be a coefficient, j is the scale of the corona, and ψ is a decaying window function of the form:

$$\psi(r; r_{\text{center}}) = 1 - \frac{|r - r_{\text{center}}|}{r_{\text{center}}} \tag{7.3}$$

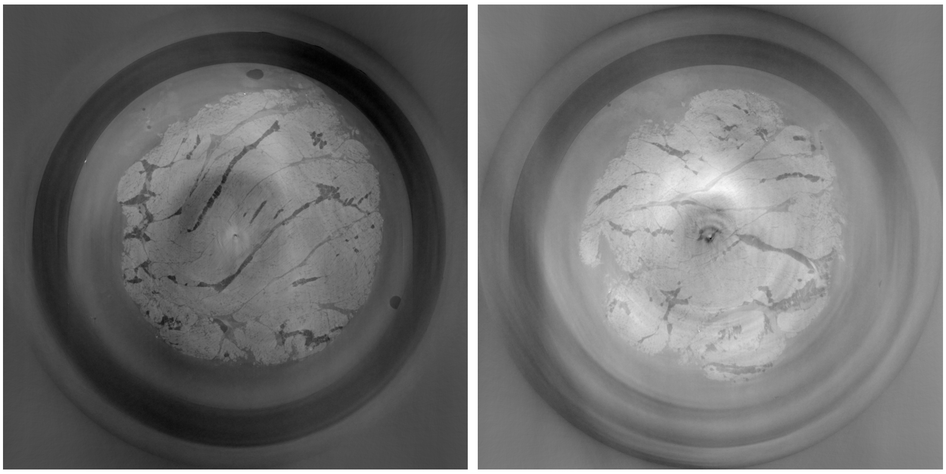
Different window functions, like Tukey, and Hann can replace ψ.

3.3 Evaluation

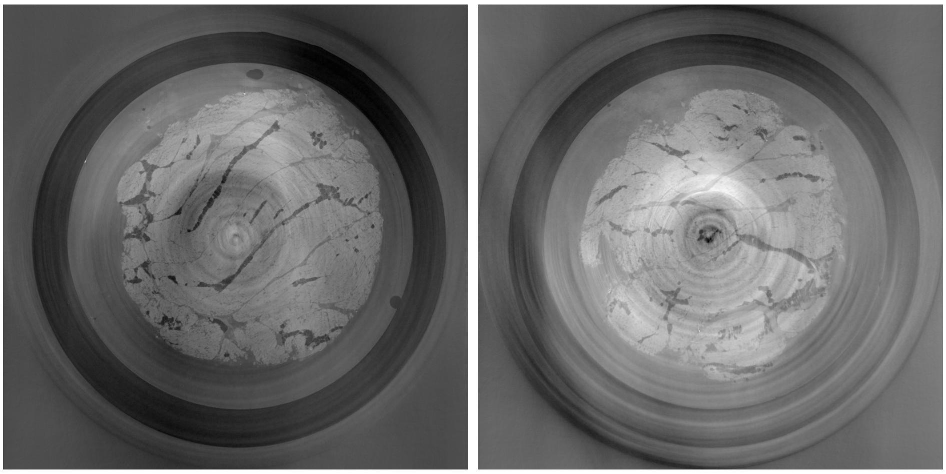
A volume of 1400 sinograms of a meat samples was used to evaluate the rings' suppression method, and the slices were constructed from the sinograms using the construction algorithm [Hsi09]. The proposed method was evaluated by visual inspection. Figures 7.6b and 7.7b show the results of some samples before and after the proposed approach was applied. A state-of-the-art method, such as the Wavelet-Fourier [MTMS09b], was applied to the same samples and the results are provided in Figures 7.6c and 7.7c. In these figures, it is very clear that the proposed method outperforms the Wavelet-Fourier approach. The proposed method is very robust to both single and band rings artifacts.



(a) Two slices with rings artifacts.

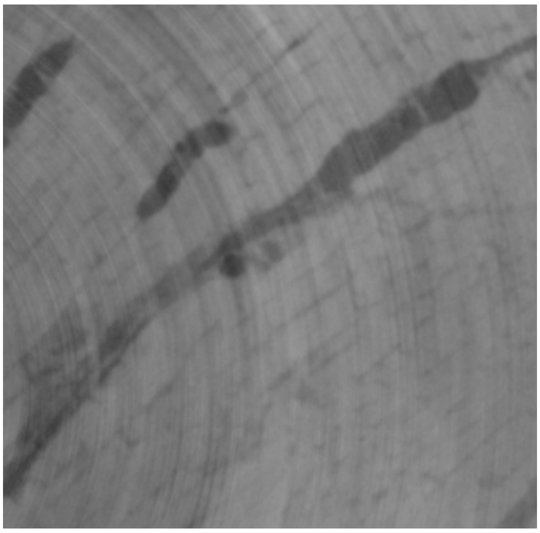


(b) Processed with the proposed approach.

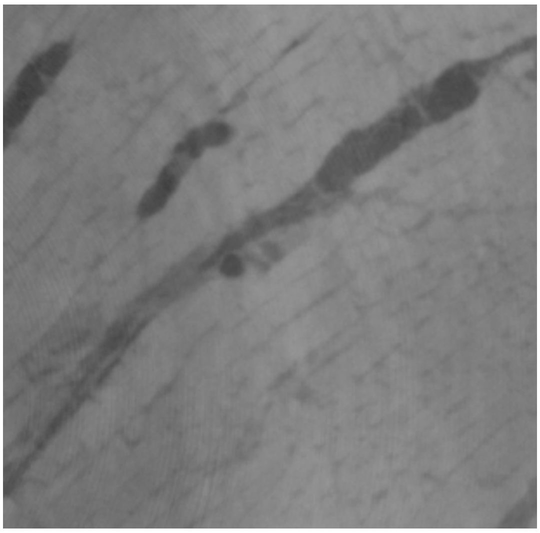


(c) Processed with the Wavelet-Fourier [MTMS09b].

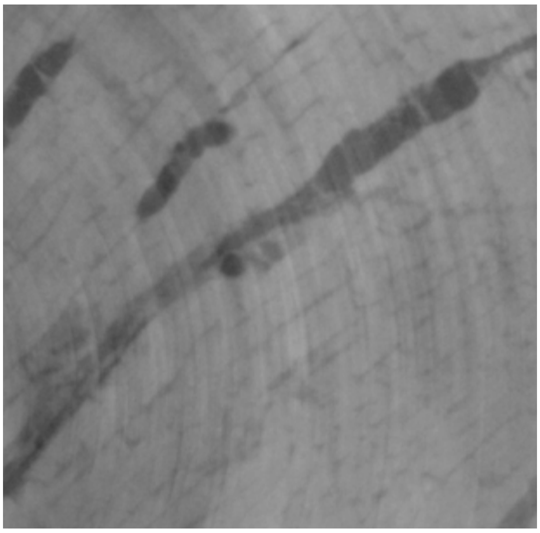
Figure 7.6: Rings' suppression results.



(a) Section of Figure. 3.1a



(b) Processed with the proposed approach.



(c) Processed with the Wavelet-Fourier [MTMS09b].

Figure 7.7: Rings' suppression results in different scales.

4 *In situ* Microfilariae Analysis

In this section, an *in situ* application for microfilariae detection (Figure 7.8), based on HPC microfilariae detection approach presented in Chapter 6, is presented.

The application is designed with a friendly Graphical User Interface (GUI). In addition, the application has several other tools for basic image preprocessing, such as image cropping and scaling. This application is developed on top of the Netbeans platform using the Java programming language, and the image processing tasks are implemented on top of the popular OpenCV library.

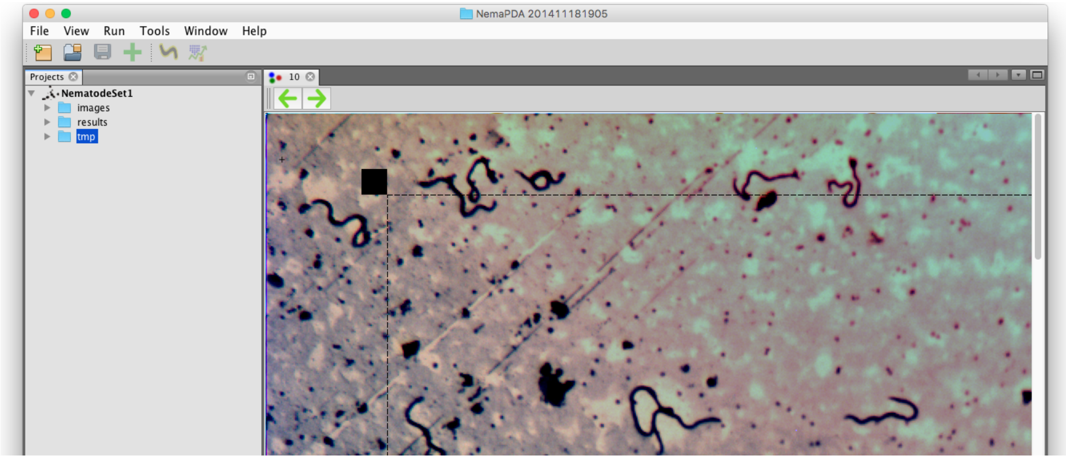


Figure 7.8: The GUI of the *In situ* microfilariae detector. The image editor is shown in the middle, and the projects' panel is shown on the left. On top, the menu and toolbar are also shown.

The workflow of this application is quite simple. The user creates a project, and imports the microfilariae images. The application then performs the detection of possible objects. Thereafter, the user selects some reference objects (Figure 7.9) and the application will filter out the unwanted artifacts based on some features such as similarities between the reference and the possible objects.

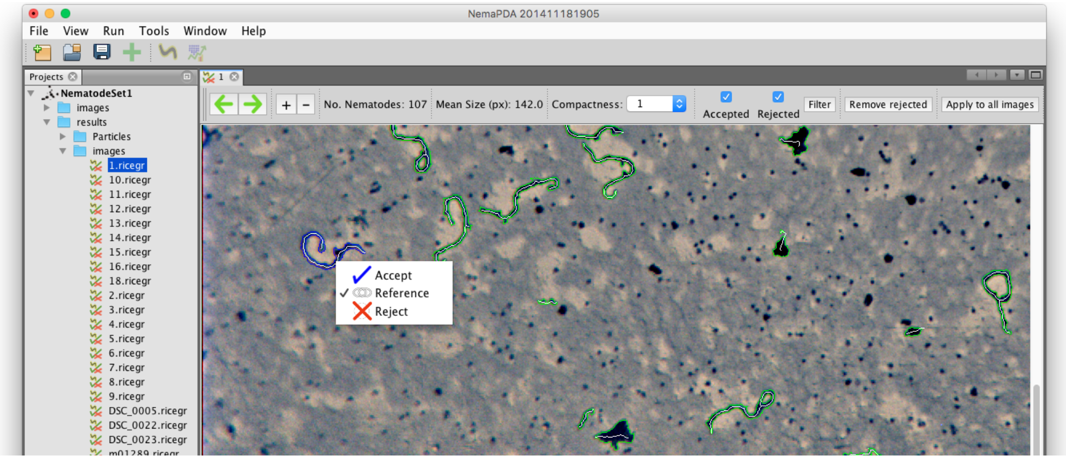


Figure 7.9: The selection of reference microfilariae. The commands on the top of the middle panel allow the users to select more parameters and apply the filtering to all images in the project.

For each candidate object (shown in green in Figure 7.9), the HPC and the color histogram of the opponent color space are calculated, resulting in two separate feature vectors for each object representing the shape and color. HPC is compared to all reference microfilariae's HPCs using the χ^2 distance, producing a score. The same is performed with the color vector producing another score. The product of these two scores is then thresholded to reject debris. In addition, the user is able to manually post-process the application's results when required. The results can be stored in a Comma Separated Value (CSV) file, which provides summary and details of the number of objects and their lengths.

8

CONCLUSION

THIS thesis presents many valuable contributions to the microscopy image processing field. A very effective method for illumination correction of images is proposed. The method is a great asset for better visualization and processing of microscopy images. Since the main focus of this work was to detect microfilariae in 2-D images, a new detection system is also presented. It is an assist to diagnosing and treating common filariasis which is, very frequently, associated with many serious health problems in the Central Africa. This system features an original hypothesis generator, and a novel powerful descriptor for microfilariae. In addition, during the work of this thesis, several methods and applications were developed to process many different image modalities.

In what the illumination correction method is concerned, the main problem, however, was centered in the estimation of the illumination bias without modeling the objects of the image into the estimated bias field. Such problem needs a careful strategy when both, the right pixels from the image and the proper bias estimator are being selected. The proposed approach is based on avoiding common problems existing in the literature. In the proposed method, the image is sampled before estimating the bias. This allows using appropriate pixels in the estimation task. The bias estimator is designed in a such way that, the bias field is iteratively made uniform. Experimental results show better quantitative and qualitative properties of the proposed approach when compared to state-of-the-art methods. In the future, this work will be applied to 3-D volumes once a suitable dataset is made available.

For microfilariae detection, basic approaches were evaluated, like segmentation followed by binary object analysis. The preliminary tests showed that, the artifacts existing in the microfilariae images along with the elasticity of the microfilariae, make the detection problem extremely difficult. Therefore, a comprehensive review of object descriptors/detectors was conducted. In addition, thin object detection approaches were also reviewed. Experimental evaluations of the existing methods, in both generic and thin objects detection, made it clear that a new descriptor needed to be designed, the Hessian Polar Context (HPC).

HPC is a local descriptor of microfilariae that treats the microfilaria as a local structure spanning a medial line. This descriptor is modeled in the polar domain to integrate both the orientation and local width of the microfilaria. The HPC is applied to object candidates proposed by a simple, yet efficient, hypothesis candidate generator developed in this work. The hypothesis generator is compared to one of the most successful state-of-the-art methods, Selective Search. Although Selective Search is shown to be a good choice, it produces unwanted objects. On the other hand, the proposed approach eliminates these objects without sacrificing the accuracy of the results.

The experiments performed in the context of detection were conducted in three stages. First, HPC is compared to very well-established object descriptors, like SIFT and LIOP. In the second stage, color information is incorporated in the tests to evaluate the importance of color in all feature descriptors. The results showed that, merging the results of two different classifiers, one for the structure and another for the color produces better results than concatenating all the features. Overall results show that HPC is superior to the other tested methods. The last stage consisted in comparing the HPC detection system to part-based HOG, a generic state-of-the-art object detector. The detection results show that, the proposed system outperforms the part-based HOG by presenting less problems, and better accuracy.

Although HPC has shown to be powerful for microfilariae description, it was not tested on other thin objects, since it is beyond the scope of this thesis. Moreover, further research is currently being conducted, and promising results indicate that, a new powerful descriptor based on HPC for these objects, will soon be ready.

Furthermore, this thesis presented a powerful method for rings' suppression in phase-contrast CT volumes. The method takes advantage of the high sparsity of the curvelets to detect the rings in the sinograms. The results of the proposed method are very encouraging. When compared to a state-of-the-art method, it showed very distinctive results. In fact, this method, is still under development within a co-operative research with the DTU University in Denmark, and an article is under preparation. We hope that, the proposed method will replace the current state-of-the-art methods, when successfully applied to modalities, other than phase-contrast.

APPENDIX A

INTEGRAL IMAGE

For fast calculation of rectangular blocks on a grid, Crow developed an approach [Cro84], that is incorporated in [VJ04] to create a feature detector for face recognition. Since then, this approach is used in many other works, such as in the popular SURF method [BTVG06]. The idea of integral image is to decrease the computation needed for block processing in a grid. For instance, the calculation of a local mean filter of a 2-D image requires sliding a window over each pixel and calculating the mean of each overlap of the window and the image. Such calculation is too slow for a simple task. On the other hand, when using the integral image, each pixel will contain the sum of all pixels in the rectangle bounded by the pixel at (0,0) and the pixel itself (Figure A.1). This way, one is able to, efficiently, perform a vast range of statistical calculations (based on the sum) for any region in the image. The sum of any region can be found by simple calculations of the sums of the region's neighbors.

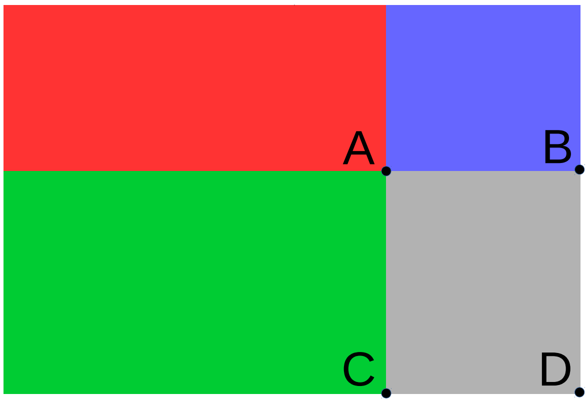


Figure A.1: Integral image calculation. A = sum( ), B = A + sum( ), C = A + sum( ) , and D = B + C - A + sum( ).

Formally, let $s_{x,y}$ be the pixel value of the image $\boldsymbol{S}$ at point (x, y), the corresponding pixel $z_{x,y}$ in the integral image $\boldsymbol{Z}$ can be recursively calculated as follows:

$$z_{x,y} = z_{x-1,y} + c_{x,y} \tag{A.1}$$

where,

$$c_{x,y} = c_{x,y-1} + s_{x,y} \tag{A.2}$$

is the cumulative row sum. For mathematical convenience, $c_{x,-1}$ and $z_{-1,y}$ are set to 0.

APPENDIX B

FEATURES POOLING USING BAG-OF-WORDS

In order to generate a feature vector for SIFT calculated on a dense grid on an image, one needs to pool the features with a fixed length vector. An approach to do that, is to create a bag-of-words [CDF+04]. It inherits its name from text processing [Har54], and can be generated for dense SIFT as follows:

1. Divide the input image into a non-overlapped set of sub-images.
2. Apply SIFT to each sub-image. Each keypoint will have a 128 dimensional feature vector.
3. Encode these vectors to create a codebook by simply applying k-means clustering with a chosen k. Each image will produce a matrix $\boldsymbol{V}^{(i)}$ ($i \leq n$ and n is

the number of images used to create the codeword) of size $128 \times m$, where m is the number of keypoints gathered from the image. The input to k-means is, therefore, a big matrix $\boldsymbol{V}^{(\text{all})}$ created by the horizontal concatenation of $\boldsymbol{V}^{(i)}$, $\forall i$. The output of k-means is a matrix $\boldsymbol{C}$ with size $128 \times k$.

In order to calculate a feature vector for each image, the following procedure is used:

1. Create a histogram vector $\boldsymbol{h}$ of size k and initialize it to zeros.

2. Apply dense SIFT as in step 2 of stage 1 (in the BoW quantization).

3. For each keypoint's vector find the index of its "best match" vector in the codebook matrix $\boldsymbol{C}$.

4. Increase the corresponding bin in $\boldsymbol{h}$ by 1.

After calculating $\boldsymbol{h}$ for all keypoint's vector, $\boldsymbol{h}$ is normalized.

BIBLIOGRAPHY

[AB76] Gerald Agin and Thomas Binford. Computer description of curved objects. *IEEE Transactions on Computers*, C-25(4):439–449, April 1976.

[AB94] Rolf Adams and Leanne Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, Jun 1994.

[ADF10] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA - USA, June 2010.

[AF12] Thomas Deselaers Alexe, Bogdan and Vittorio Ferrari. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, Nov 2012.

[AGS+12] Rehan Ali, Mark Gooding, Tünde Szilágyi, Borivoj Vojnovic, Martin Christlieb, and Michael Brady. Automatic segmentation of adherent biological cell boundaries and nuclei from brightfield microscopy images. *Machine Vision and Applications*, 23(4):607–621, 2012.

[AHF06] Alaa Abdel-Hakim and A.A. Farag. CSIFT: A SIFT descriptor with color invariant characteristics. In *Computer Vision and Pattern*

Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 1978–1983, 2006.

[AHRwC10] Danni Ai, Xianhua Han, Xiang Ruan, and Yen wei Chen. Adaptive color independent components based SIFT descriptors for image classification. In *The 20th International Conference on Pattern Recognition (ICPR)*, pages 2436–2439, Istanbul, Turkey, August 2010.

[AMFM09] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. From contours to regions: An empirical evaluation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2294–2301, Miami, FL - USA, June 2009.

[AMFM11] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011.

[AMHP09] Timo Ahonen, Jiří Matas, Chu He, and Matti Pietikäinen. Rotation invariant image description with local binary pattern histogram fourier features. In Arnt-Børre Salberg, JonYngve Hardeberg, and Robert Jenssen, editors, *Image Analysis*, volume 5575 of *Lecture Notes in Computer Science*, pages 61–70. Springer Berlin Heidelberg, 2009.

[APTB+14] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 328–335, Columbus, OH - USA, June 2014.

[ARQA07] Mohammed Al-Rawi, Munib Qutaishat, and Mohammed Arrar. An improved matched filter for blood vessel detection of digital retinal images. *Computers in Biology and Medicine*, 37(2):262 – 267, 2007.

[AT12] Faroq AL-Tam. Warping, matching and reporting 2-D electrophoresis protein gel images. Master's thesis, Universidade do Algarve, 2012.

[ATAA+13] Faroq AL-Tam, Helene Adam, António dos Anjos, Mathias Lorieux, Pierre Larmande, Alain Ghesquière, Stefan Jouannic, and

Hamid Reza Shahbazkia. P-TRAP: a panicle trait phenotyping tool. *BMC Plant Biology*, 13(1):1–14, 2013.

[ATdABS15] Faroq AL-Tam, António dos Anjos, Stephane Bellafiore, and Hamid Reza Shahbazkia. Detection of root knot nematodes in microscopy images. In *Proceedings of the International Conference on Bioimaging (BIOSTEC 2015)*, pages 76–81, Lisbon, 2015.

[ATdAP+16] Faroq AL-Tam, António dos Anjos, Sébastien Pion, Michel Boussinesq, and Hamid Shahbazkia. Microfilariae classification using multiple classifiers for color and shape features. In *Open Engineering Journal*, 2016. (to appear).

[BB09] Wilhelm Burger and Mark J Burge. *Digital image processing: an algorithmic introduction using Java.* Springer Science & Business Media, 2009.

[Bea78] P. R. Beaudet. Rotationally invariant image operators. In *Proceedings of the 4th International Joint Conference on Pattern Recognition*, pages 579–583, Kyoto, Japan, November 1978.

[BGGWC03] Michel Boussinesq, Jacques Gardon, Nathalie Gardon-Wendel, and Jean-Philippe Chippaux. Clinical picture, epidemiology and outcome of loa-associated serious adverse events related to mass ivermectin treatment of onchocerciasis in cameroon. *Filaria Journal*, 2(Suppl 1), 2003.

[BGV92] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[BHW10] Asa Ben-Hur and Jason Weston. A user's guide to support vector machines. In Oliviero Carugo and Frank Eisenhaber, editors, *Data Mining Techniques for the Life Sciences*, volume 609 of *Methods in Molecular Biology*, pages 223–239. Humana Press, 2010.

[BL02] Matthew Brown and David Lowe. Invariant features from interest point groups. In *Proceedings of the British Machine Vision Conference*, pages 23.1–23.10, Cardiff, 2002. BMVA Press.

[BLL07] Xiang Bai, Longin Jan Latecki, and Wen-Yu Liu. Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):449–462, March 2007.

[BMP02] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522, Apr 2002.

[Bou06] Michel Boussinesq. Loiasis. *Annals of tropical medicine and parasitology*, 100(8):715–731, 2006.

[BSI08] Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 1–8, Anchorage, AL - USA, June 2008.

[BSP04] Ravi Bansal, Lawrence H Staib, and Bradley S Peterson. Correcting nonuniformities in MRI intensities using entropy minimization based on an elastic model. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*, pages 78–86. Springer, 2004.

[BT00] Michael Buckley and Hugues Talbot. Flexible linear openings and closings. In John Goutsias, Luc Vincent, and DanS. Bloomberg, editors, *Mathematical Morphology and its Applications to Image and Signal Processing*, volume 18 of *Computational Imaging and Vision*, pages 109–118. Springer US, 2000.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision - ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin Heidelberg, 2006.

[BVZ01] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, November 2001.

[Can86] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov 1986.

[CCK+89] Subhasis Chaudhuri, Shankar Chatterjee, Norman Katz, Mark Nelson, and Michael Goldbaum. Detection of blood vessels in retinal images using two-dimensional matched filters. *IEEE Transactions on Medical Imaging*, 8(3):263–269, Sep 1989.

[CD00] Emmanuel Candes and David Donoho. Curvelets - a surprisingly effective non adaptive representation for objects with edges. In *Curves and Surface Fitting, A. Cohen, C. Rabut, and L. L. Schumaker, Eds. Nashville, TN: Vanderbilt Univ. Press*, 2000.

[CDF+04] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[CG10] Michael Crosier and Lewis Griffin. Using basic image features for texture classification. *International Journal of Computer Vision*, 88(3):447–460, 2010.

[CL11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[CLR+01] Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein, et al. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.

[CLSF10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 778–792, Berlin, Heidelberg, 2010. Springer-Verlag.

[CM02] Dorin Comaniciu and Peter Meer. Cell image segmentation for diagnostic pathology. In JasjitS. Suri, S.Kamaledin Setarehdan, and

Sameer Singh, editors, *Advanced Algorithmic Approaches to Medical Image Segmentation*, Advances in Computer Vision and Pattern Recognition, pages 541–558. Springer London, 2002.

[CR03] Haili Chui and Anand Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2-3):114 – 141, 2003.

[CRCG11] Angel Cruz-Roa, Juan Caicedo, and Fabio González. Visual pattern mining in histology image collections using bag of features. *Artificial intelligence in medicine*, 52(2):91–106, 2011.

[Cro84] Franklin Crow. Summed-area tables for texture mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 207–212, New York, NY, USA, 1984. ACM.

[CS02] Sung-Hyuk Cha and Sargur Srihari. On measuring the distance between histograms. *Pattern Recognition*, 35(6):1355 – 1370, 2002.

[CTS+11] Huizhong Chen, Sam Tsai, Georg Schroth, David Chen, Radek Grzeszczuk, and Bernd Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In *The 18th IEEE International Conference on Image Processing (ICIP)*, pages 2609–2612, Sept 2011.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[CV01] Tony Chan and Luminita Vese. Active contours without edges. *Trans. Img. Proc.*, 10(2):266–277, February 2001.

[CWB08] Emmanuel Candes, Michael Wakin, and Stephen Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

[CYV00] Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, Sep 2000.

[CYZ+05] Terrence Chen, Wotao Yin, Xiang Sean Zhou, Dorin Comaniciu, and Thomas S Huang. Illumination normalization for face recognition and uneven background correction using total variation based image models. In *Computer Vision and Pattern Recognition*, volume 2, pages 532–539. IEEE, 2005.

[dAME+11] António dos Anjos, Anders LB Møller, Bjarne K Ersbøll, Christine Finnie, and Hamid R Shahbazkia. New approach for segmentation and quantification of two-dimensional gel electrophoresis images. *Bioinformatics*, 27(3):368–375, 2011.

[dAS08] António dos Anjos and Hamid Shahbazkia. Bi-level image thresholding - A fast method. In *Proceedings of the First International Conference on Biomedical Electronics and Devices, BIOSIGNALS 2008, Funchal, Madeira, Portugal, January 28-31, 2008, Volume 2*, pages 70–76, 2008.

[DB06] M. Donoser and H. Bischof. 3D segmentation by maximally stable volumes (MSVs). In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 63–66, 2006.

[DBB02] Hans Du Buf and Micha M Bayer. *Automatic diatom identification*, volume 51. World Scientific, 2002.

[DDS+09] Jia Deng, Wei Dong, Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL - USA, June 2009.

[Dij59] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition.*, volume 1, pages 886–893 vol. 1, San Diego, CA - USA, June 2005.

[EGW+10] Mark Everingham, Luc Gool, Christopher Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vision*, 88(2):303–338, June 2010.

[Elk03] Charles Elkan. Using the triangle inequality to accelerate k-means. In *ICML*, volume 3, pages 147–153, 2003.

[ENM+14] Hildur Einarsdóttir, Mikkel Schou Nielsen, Rikke Miklos, René Lametsch, Robert Feidenhans'l, Rasmus Larsen, and Bjarne Kjaer Ersboll. Analysis of micro-structure in raw and heat treated meat emulsions from multimodal x-ray microtomography. *Innovative Food Science & Emerging Technologies*, 24:88 – 96, 2014. Food Microstructure.

[FFJS08] Vittorio Ferrari, Loic Fevrier, Frederic Jurie, and Cordelia Schmid. Groups of adjacent contour segments for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):36–51, Jan 2008.

[FFP05] Fei-Fei and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition,*, volume 2, pages 524–531 vol. 2, San Diego, CA - USA, June 2005.

[FGMR10] Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[FH04] Pedro Felzenszwalb and Daniel Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, September 2004.

[FJS07] Vittorio Ferrari, Frederic Jurie, and Cordelia Schmid. Accurate object detection with deformable shape models learnt from images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN - USA, June 2007.

[FL95] Mario Figueiredo and Jose Leitao. A nonsmoothing approach to the estimation of vessel contours in angiograms. *IEEE Transactions on Medical Imaging*, 14(1):162–172, 1995.

[FNVV98] Alejandro. Frangi, Wiro Niessen, Koen Vincken, and Max Viergever. Multiscale vessel enhancement filtering. In WilliamM. Wells, Alan

Colchester, and Scott Delp, editors, *Medical Image Computing and Computer-Assisted Interventation - MICCAI-98*, volume 1496 of *Lecture Notes in Computer Science*, pages 130–137. Springer Berlin Heidelberg, 1998.

[For07] Per-Erik Forssén. Maximally stable colour regions for recognition and matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN - USA, June 2007.

[FtHRKV92] Luc Florack, Bart ter Haar Romeny, Jan Koenderink, and Max Viergever. Scale and the differential structure of images. *Image and Vision Computing*, 10(6):376–388, 1992.

[FTVG06] Vittorio Ferrari, Tinne Tuytelaars, and Luc Van Gool. Object detection by contour segment networks. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part III*, ECCV'06, pages 14–28, Berlin, Heidelberg, 2006. Springer-Verlag.

[FWH12] Bin Fan, Fuchao Wu, and Zhanyi Hu. Rotationally invariant descriptors using intensity order pooling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):2031–2045, Oct 2012.

[GDDM14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587, Columbus, OH - USA, June 2014.

[GGWDN+97] Jacques Gardon, Nathalie Gardon-Wendel, Demanga-Ngangue, Joseph Kamgno, Jean-Philippe Chippaux, and Michel Boussinesq. Serious reactions after mass treatment of onchocerciasis with ivermectin in an area endemic for loa loa infection. *The Lancet*, 350(9070):18 – 22, 1997.

[GHR+14] Daniel Goehring, Judy Hoffman, Erid Rodner, Kate Saenko, and Trevor Darrell. Interactive adaptation of real-time object detectors. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1282–1289, Hong Kong - China, May 2014.

[GLAM09] Chunhui Gu, Jasmine Lim, Pablo Arbeláez, and Jagannath Malik. Recognition using regions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1030–1037, Miami, FL - USA, June 2009.

[GRB08] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. Object categorization using co-occurrence, location and appearance. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AL - USA, June 2008.

[GVL96] Gene Golub and Charles Van Loan. *Matrix Computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[GW02] Rafael Gonzales and Richard Woods. *Digital Image Processing, 2^{nd} Edition.* Prentice Hall, 2002.

[HA02] Antti Happonen and Sakari Alenius. Sinogram filtering using a stackgram domain. In *Proceedings of the Second IASTED International Conference: Visualization, Imaging and Image Processing*, pages 339–343, 2002.

[Har54] Zellig Harris. Distributional structure. *Word*, 1954.

[HBDS15] Jan Hosang, Rodrigo Benenson, Piotr Dollár, and Bernt Schiele. What makes for effective detection proposals? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[HBT04] Henk Heijmans, Michael Buckley, and Hugues Talbot. Path-based morphological openings. In *International Conference on Image Processing*, volume 5, pages 3085–3088 Vol. 5, Singapore, Oct 2004.

[HBT05] Henk Heijmans, Michael Buckley, and Hugues Talbot. Path openings and closings. *Journal of Mathematical Imaging and Vision*, 22(2-3):107–119, 2005.

[HDO+98] Marti Hearst, Susan Dumais, Edgar Osman, John Platt, and Bernhard Scholkopf. Support vector machines. *Intelligent Systems and their Applications, IEEE*, 13(4):18–28, Jul 1998.

[Hea02] Michael Heath. *Scientific computing: an introductory survey.* McGraw-Hill Higher Education. McGraw-Hill, 2002.

[HKG00] Jiří Hladůvka, Andreas König, and Eduard Gröller. Exploiting eigenvalues of the hessian matrix for volume decimation. Technical Report TR-186-2-00-19, October 2000.

[HKM+97] Jing Huang, S Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zabih. Image indexing using color correlograms. In *IEEE Computer Conference on Computer Vision and Pattern Recognition*, pages 762–768, San Juan, PR - USA, Jun 1997.

[HL02] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, Mar 2002.

[HM14] Anders Hast and Andrea Marchetti. Improved illumination correction that preserves medium sized objects. *Machine Graphics & Vision*, 23(1/2):3–20, 2014.

[HMR12] Bharath Hariharan, Jitendra Malik, and Deva Ramanan. Discriminative decorrelation for clustering and classification. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ECCV'12, pages 459–472, Berlin, Heidelberg, 2012. Springer-Verlag.

[HS88] Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[HSD73] Robert Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, (6):610–621, 1973.

[Hsi09] Jiang Hsieh. Computed tomography: principles, design, artifacts, and recent advances. SPIE Bellingham, WA, 2009.

[Jai89] Anil K Jain. *Fundamentals of digital image processing.* Prentice-Hall, Inc., 1989.

[JATD15] Thomas Martini Jørgensen, F. AL-Tam, and Vedrana Andersen Dahl. *Artifact removal in Differential Phase Contrast X-ray Computed Tomography*, pages 105–108. European Cooperation in Science and Technology, 2015.

[JRW97] Daniel Jobson, Zia-ur Rahman, and Glenn Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image Processing*, 6(7):965–976, July 1997.

[JSD+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[KJK10] Libor Kubecka, Jiri Jan, and Radim Kolar. Retrospective illumination correction of retinal images. *International journal of biomedical imaging*, 2010.

[Kno39] James Knott. A method for making microfilarial surveys on day blood. *Transactions of the Royal Society of Tropical Medicine and Hygiene*, 33(2):191 – 196, 1939.

[KQ04] Cemil Kirbas and Francis Quek. A review of vessel extraction techniques and algorithms. *ACM Computing Surveys*, 36(2):81–121, June 2004.

[KS04] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–506, Washington, D.C. USA, 2004. IEEE.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[Kun04] Ludmila I Kuncheva. Combining pattern classifiers. *Methods and Algorithms. Wiley, Chichester*, 2004.

[KWT88] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[KZBB11] Ron Kimmel, Cuiping Zhang, Alexander Bronstein, and Michael Bronstein. Are MSER features really interesting? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2316–2320, Nov 2011.

[LA08] Jing Li and Nigel M. Allinson. A comprehensive review of current local features for computer vision. *Neurocomputing*, 71(10-12):1771–1787, June 2008.

[LB01] Joakim Lindblad and Ewert Bengtsson. A comparison of methods for estimation of intensity non-uniformities in 2d and 3d microscope images of fluorescence stained cells. In *Proceedings of the 12th Scandinavian Conference on Image Analysis (SCIA)*, pages 264–271, 2001.

[LBH09] Christoph Lampert, Matthew Blaschko, and Thomas Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2129–2142, Dec 2009.

[LBU12] Stamatios Lefkimmiatis, Aurélien Bourquard, and Michael Unser. Hessian-based norm regularization for image restoration with biomedical applications. *IEEE Transactions on Image Processing*, 21(3):983–995, 2012.

[LCS11] Stefan Leutenegger, Margarita Chli, and Roland Siegwart. BRISK: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision*, pages 2548–2555, Barcelona, Spain, Nov 2011.

[Lew92] David Dolan Lewis. *Representation and learning in information retrieval*. PhD thesis, University of Massachusetts, 1992.

[LGWG09] Chunming Li, Chris Gatenby, Li Wang, and John C Gore. A robust parametric method for bias field estimation and segmentation of MR images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 218–223, Miami, FL - USA, 2009. IEEE.

[Lin94] Tony Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21:224–270, 1994.

[Lin96] Tony Lindeberg. Edge detection and ridge detection with automatic scale selection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 465–470, San Francisco, CA - USA, Jun 1996.

[Lin98] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, November 1998.

[LLLY12] Kuan Li, Zhi Lu, Wenyin Liu, and Jianping Yin. Cytoplasm and nucleus segmentation in cervical smear images using radiating {GVF} snake. *Pattern Recognition*, 45(4):1255 – 1264, 2012.

[LLS92] Louisa Lam, Seong-Whan Lee, and Ching Suen. Thinning methodologies- a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, Sep 1992.

[LMVP00] Boštjan Likar, Josephus Bernard Antoine Maintz, Max Viergever, and Franjo Pernus. Retrospective shading correction based on entropy minimization. *Journal of Microscopy*, 197(3):285–295, 2000.

[LOD12] Conor Leahy, Andrew O'Brien, and Chris Dainty. Illumination correction of retinal images using laplace interpolation. *Applied optics*, 51(35):8383–8389, 2012.

[Low99] David Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, Kerkyra, Greece, 1999.

[Low04] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[LSP05] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, August 2005.

[LSP06] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, New York City, NY - USA, 2006.

[LSS+14] Kun Liu, Henrik Skibbe, Thorsten Schmidt, Thomas Blein, Klaus Palme, Thomas Brox, and Olaf Ronneberger. Rotation-invariant hog descriptors using fourier analysis in polar and spherical coordinates. *International Journal of Computer Vision*, 106(3):342–364, 2014.

[LY08] Guang-Hai Liu and Jing-Yu Yang. Image retrieval based on the texton co-occurrence matrix. *Pattern Recognition*, 41(12):3521 – 3527, 2008.

[MC06] Ana Maria Mendonca and Aurelio Campilho. Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction. *IEEE Transactions on Medical Imaging*, 25(9):1200–1213, 2006.

[MCUP04] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767, 2004. British Machine Vision Computing 2002.

[MDD14] Vincent Morard, Petr Dokládal, and Etienne Decenciere. Parsimonious path openings and closings. *IEEE Transactions on Image Processing*, 23(4):1543–1555, April 2014.

[MFM02] David Martin, Charless Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using brightness and texture. In *Advances in Neural Information Processing Systems*, pages 1255–1262, 2002.

[Mig08] Max Mignotte. Segmentation by fusion of histogram-based K-Means clusters in different color spaces. *IEEE Transactions on Image Processing*, 17(5):780–787, May 2008.

[MJS+04] Erik Meijering, Mathews Jacob, JCF Sarria, Pl Steiner, H Hirling, and Michael Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry Part A*, 58(2):167–176, 2004.

[ML11] Tianyang Ma and Longin Jan Latecki. From partial shape matching through local deformation to robust global shape similarity for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1441–1448, Colorado Springs, CO - USA, June 2011.

[MMOC11] Wenye Ma, Jean-Michel Morel, Stanley Osher, and Aichi Chien. An ℓ_1-based variational model for retinex theory and its application to medical images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 153–160, Colorado Springs, CO - USA, 2011. IEEE.

[Mor77] Hans. Morevec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'77, pages 584–584, San Francisco, CA, USA, 1977. Morgan Kaufmann Publishers Inc.

[MRH02] Arnold Meijster, Jos BTM Roerdink, and Wim H Hesselink. A general algorithm for computing distance transforms in linear time. In *Mathematical Morphology and its applications to image and signal processing*, pages 331–340. Springer, 2002.

[MS89] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on pure and applied mathematics*, 42(5):577–685, 1989.

[MS02] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, pages 128–142, London, UK, 2002. Springer-Verlag.

[MS04] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, October 2004.

[MS05] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct 2005.

[MTMS09a] Beat Münch, Pavel Trtik, Federica Marone, and Marco Stampanoni. Stripe and ring artifact removal with combined wavelet — fourier filtering. *Opt. Express*, 17(10):8567–8591, May 2009.

[MTMS09b] Beat Münch, Pavel Trtik, Federica Marone, and Marco Stampanoni. Stripe and ring artifact removal with combined wavelet—fourier filtering. *Opt. Express*, 17(10):8567–8591, May 2009.

[MTS+05] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International journal of computer vision*, 65(1-2):43–72, 2005.

[NJT06] Eric Nowak, Frédéic Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In Aleš̃ Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision - ECCV 2006*, volume 3954 of *Lecture Notes in Computer Science*, pages 490–503. Springer Berlin Heidelberg, 2006.

[NS08] David Nistér and Henrik Stewénius. Linear time maximally stable extremal regions. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision - ECCV 2008*, volume 5303 of *Lecture Notes in Computer Science*, pages 183–196. Springer Berlin Heidelberg, 2008.

[OF96] Bruno A Olshausen and David J Field. Natural image statistics and efficient coding*. *Network: computation in neural systems*, 7(2):333–339, 1996.

[OPM02] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, Jul 2002.

[Org97] World Health Organization. Bench aids for the diagnosis of filarial infections. 1997.

[Ort12] Raphael Ortiz. FREAK: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 510–517, Washington, DC, USA, 2012. IEEE Computer Society.

[Ots75] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[PD07] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN - USA, June 2007.

[Pla99] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[PM90] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[PW96] Ashley Walker Perkins, Simon and Erik Wolfart. *Hypermedia Image Processing Reference*. J. Wiley & Sons Publishing, 1996.

[RBK96] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Human face detection in visual scenes. In D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 875–881. MIT Press, 1996.

[RC78] TW Ridler and S Calvard. Picture thresholding using an iterative selection method. *IEEE Transactions on Systems, Man and Cybernetics*, 8(8):630–632, 1978.

[RC07] Ch Rambabu and Indrajit Chakrabarti. An efficient immersion-based watershed transform method and its prototype architecture. *Journal of Systems Architecture*, 53(4):210 – 226, 2007.

[RDB10] Hayko Riemenschneider, Michael Donoser, and Horst Bischof. Using partial edge contour matches for efficient object category localization. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision - ECCV 2010*, volume 6315 of *Lecture Notes in Computer Science*, pages 29–42. Springer Berlin Heidelberg, 2010.

[RLC+10] Riklin Raviv, Vebjorn Ljosa, Annie Conery, Frederick Ausubel, Anne Carpenter, Polina Golland, and Carolina Wählby. Morphology-guided graph search for untangling objects: C. elegans analysis. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI*, pages 634–641. Springer, 2010.

[ROF92] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[RRKB11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2564–2571, Washington, DC, USA, 2011. IEEE Computer Society.

[SB91] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[SC05] Frank Shih and Shouxian Cheng. Automatic seeded region growing for color image segmentation. *Image and Vision Computing*, 23(10):877–886, September 2005.

[Ser83] Jean Serra. *Image analysis and mathematical morphology*. Academic Press, Inc., Orlando, FL, USA, 1983.

[SHDB89] Michael Spann, C Horne, and JM Hans Du Buf. The detection of thin structures in images. *Pattern Recognition Letters*, 10(3):175 – 179, 1989.

[SMB00] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, June 2000.

[SMF10] Jean-Luc Starck, Fionn Murtagh, and Jalal M Fadili. *Sparse image and signal processing: wavelets, curvelets, morphological diversity.* Cambridge university press, 2010.

[SS04] Mehmet Sezgin and Bülent Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–168, 2004.

[ST01] Pierre Soille and Hugues Talbot. Directional morphological filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1313–1329, Nov 2001.

[Ste98] Carsten Steger. An unbiased detector of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):113–125, 1998.

[SWKJ02] Alexei Samsonov, Ross Whitaker, Eugene Kholmovski, and Chris Johnson. Parametric method for correction of intensity inhomogeneity in MRI data. *International Society for Magnetic Resonance in Medicine*, page 154, 2002.

[SZE98] John Sled, Alex Zijdenbos, and Alan Evans. A nonparametric method for automatic correction of intensity nonuniformity in mri data. *IEEE Transactions on Medical Imaging*, 17(1):87–97, 1998.

[TJW08] Tolga Tasdizen, Elizabeth Jurrus, and Ross T Whitaker. Nonuniform illumination correction in transmission electron microscopy. In *MICCAI Workshop on Microscopic Image Analysis with Applications in Biology*, pages 5–6, 2008.

[TLP02] Dejan Tomaževič, Boštjan Likar, and Franjo Pernuš. Comparative evaluation of retrospective shading correction methods. *Journal of Microscopy*, 208(3):212–223, 2002.

[TM08] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: A survey. *Found. Trends. Comput. Graph. Vis.*, 3(3):177–280, July 2008.

[TMdB15] Kasim Terzić, Hussein Adnan Mohammed, and J.M.H. du Buf. Shape detection with nearest neighbour contour fragments. In Mark

W. Jones Xianghua Xie and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 59.1–59.12, Swansea - UK, September 2015. BMVA Press.

[TTD08] Olena Tankyevych, Hugues Talbot, and Petr Dokladal. Curvilinear morpho-hessian filter. In *The 5th IEEE International Symposium on Biomedical Imaging: From Nano*, pages 1011–1014, May 2008.

[TVW02] Alexandru Telea and Jarke J Van Wijk. An augmented fast marching method for computing skeletons and centerlines. In *Proceedings of the symposium on Data Visualisation 2002*, pages 251–ff. Eurographics Association, 2002.

[UvdSGS13] Jasper Uijlings, Koen van de Sande, Theo Gevers, and Arnold Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[vdSGS10] Koen van de Sande, Theo Gevers, and Cees Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, Sept 2010.

[VF10] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 1469–1472, New York, NY, USA, 2010. ACM.

[Vin00] Luc Vincent. Granulometries and opening trees. *Fundamenta Informaticae*, 41(1-2):57–90, 2000.

[VJ04] Paul Viola and MichaelJ. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[VK05] Vladimir Vezhnevets and Vadim Konouchine. "GrowCut" - Interactive Multi-Label ND Image Segmentation By Cellular Automata. In *international conference on Computer Graphics and Vision.* Novosibirsk Akademgorodok, Russia, 2005.

[VPL07] Uros Vovk, Franjo Pernus, and Boštjan Likar. A review of methods for correction of intensity inhomogeneity in MRI. *IEEE Transactions on Medical Imaging*, 26(3):405–421, 2007.

[VS91] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, Jun 1991.

[WFW11] Zhenhua Wang, Bin Fan, and Fuchao Wu. Local intensity order pattern for feature description. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 603–610, Nov 2011.

[WMC10] Qiang Wu, Fatima Merchant, and Kenneth Castleman. *Microscope image processing*. Academic press, 2010.

[YCS00] Peter Yim, Peter Choyke, and Ronald Summers. Gray-scale skeletonization of small vessels in magnetic resonance angiography. *IEEE Transactions on Medical Imaging*, 19(6):568–576, 2000.

[YO12] Pradeep Yarlagadda and Björn Ommer. From meaningful contours to discriminative object shape. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part I*, ECCV'12, pages 766–779, Berlin, Heidelberg, 2012. Springer-Verlag.

[You01] Ian T. Young. *Shading Correction: Compensation for Illumination and Sensor Inhomogeneities*. John Wiley & Sons, Inc., 2001.

[YYH93] Hiromitsu Yamada, Kazuhiko Yamamoto, and Katsumi Hosokawa. Directional mathematical morphology and reformalized hough transformation for the analysis of topographic maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):380–387, Apr 1993.

[ZD14] C.Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014*, volume 8693 of *Lecture Notes in Computer Science*, pages 391–405. Springer International Publishing, 2014.

[ZG10] Yuanjie Zheng and James Gee. Estimation of image bias field with sparsity constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 255–262, Colorado Springs, CO - USA, 2010. IEEE.

[ZGAG09] Yuanjie Zheng, Murray Grossman, Suyash P Awate, and James C Gee. Automatic correction of intensity nonuniformity from sparseness of gradient distribution in medical images. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2009*, pages 852–859. Springer, 2009.

[ZK01] Frederic Zana and Jean-Claude Klein. Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation. *IEEE Transactions on Image Processing*, 10(7):1010–1019, 2001.

[ZLK+09] Yuanjie Zheng, Stephen Lin, Chandra Kambhamettu, Jingyi Yu, and Sing Bing Kang. Single-image vignetting correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2243–2256, 2009.

[ZS84] Tongjie Zhang and Ching Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, March 1984.

[ZVFR15] Xiangxin Zhu, Carl Vondrick, CharlessC. Fowlkes, and Deva Ramanan. Do we need more training data? *International Journal of Computer Vision*, pages 1–17, 2015.

[ZVRF12] Xiangxin Zhu, Carl Vondrick, Deva Ramanan, and Charless Fowlkes. Do we need more training data or better models for object detection? In *British Machine Vision Conference (BMVC)*, Surrey - UK, 2012.

[ZYCA06] Qiang Zhu, M.-C. Yeh, Kwang-Ting Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498, 2006.

[ZZZK10] Bob Zhang, Lin Zhang, Lei Zhang, and Fakhri Karray. Retinal vessel extraction by matched filter with first-order derivative of gaussian. *Computers in biology and medicine*, 40(4):438–445, 2010.