Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

τομεας σημάτων, ελεγχου και ρομποτικης

Ανίχνευση συμμετρίας βασισμένη στην εκμάθηση μηχανών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σταύρος Τσόγκας

Επιβλέποντες: Πέτρος Α. Μαραγκός
Καθηγητής Ε.Μ.Π.,

Ιάσονας Κόκκινος
Επίκουρος καθηγητής ECP

Αθήνα, Οκτώβριος 2011

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Σηματων, Ελεγχου και Ρομποτικης

Ανίχνευση συμμετρίας βασισμένη στην εκμάθηση μηχανών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σταύρος Τσόγκας

Επιβλέποντες: Πέτρος Α. Μαραγκός
Καθηγητής Ε.Μ.Π.,

Ιάσονας Κόκκινος
Επίκουρος καθηγητής ECP

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 27 Οκτωβρίου 2011.

.....	.....	.....
Πέτρος Μαραγκός Καθηγητής Ε.Μ.Π.	Ιάσονας Κόκκινος Επίκουρος καθηγητής ECP	Γεώργιος Καραγιάννης Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2011

.....
Σταύρος Τσόγκας
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Abstract

In this diploma thesis we introduce a novel framework for detecting ridges and bilateral symmetry in natural images using supervised learning. Since there is no existing ground truth dataset for ridges, we begin by constructing a preliminary one automatically, using images from the Berkeley segmentation dataset(BSDS300). Motivated by the work of Martin et al. on boundary detection, we use different combinations of low-level brightness, color and texture cues, collected at multiple scales and orientations, to train a ridge classifier. The learning algorithms we consider are logistic regression and multiple instance learning, and the training data consist of natural images taken from the Berkeley segmentation dataset.

For the evaluation of our method we use precision-recall curves. Qualitative and quantitative results for the various algorithms and feature combinations used are presented. We also compare our results to those obtained by the ridge detection with automated scale selection algorithm by Lindeberg, and we observe that our approach performs better. Finally, we discuss possible higher-level applications where our method could prove useful as a front-end step.

Key words: computer vision, machine learning, classifier, ridge detection, symmetry, features, ground truth, evaluation, logistic regression, multiple instance learning, precision, recall, training.

Περίληψη

Σε αυτή τη διπλωματική εργασία εξερευνούμε το πρόβλημα της ανίχνευσης αξονων συμμετριας και κορυφογραμμων (ridges) σε εικονες, από μία νέα σκοπιά, χρησιμοποιώντας επιβλεπόμενη μάθηση (supervised learning). Εφόσον δεν υπάρχει κάποιο σύνολο δεδομένων επαλήθευσης (ground truth) για κορυφογραμμές, αρχικά κατασκευάζουμε ένα για προκαταρκτική χρήση, βασιζόμενοι στο σύνολο δεδομένων κατάτμησης του Berkeley (Berkeley segmentation dataset–BSDS300). Ακολουθώντας τα βήματα των Martin et al. για ανίχνευση συνόρων, χρησιμοποιούμε διαφορετικούς συνδυασμούς χαρακτηριστικών φωτεινότητας, χρώματος και υφής, τα οποία συλλέγονται σε πολλαπλές κλίμακες και κατευθύνσεις, για την εκμάθηση του ανιχνευτή κορυφογραμμων. Οι αλγοριθμοι εκμάθησης που χρησιμοποιούμε είναι λογιστική παλινδρόμηση (logistic regression) και εκμάθηση από πολλά στιγμιότυπα (multiple instance learning, ενώ το σύνολο δεδομένων εκπαίδευσης αποτελείται από φυσικές εικόνες του BSDS300.

Για την αξιολόγηση της μεθόδου μας, χρησιμοποιούμε καμπύλες ακρίβειας-επανάκλησης (precision-recall curves). Επιπρόσθετα, παρουσιάζονται ποιοτικά και ποσοτικά αποτελέσματα για τους διαφορετικούς αλγορίθμους και συνδυασμούς χαρακτηριστικών που χρησιμοποιούνται. Συγκρίνουμε επίσης τα αποτελέσματά μας με αυτά που δίνει η εφαρμογή ανίχνευσης κορυφογραμμών με χρήση της μεθόδου με αυτόματη επιλογή κλίμακας, του Lindeberg, και παρατηρούμε ότι η προσέγγισή μας αποδίδει καλύτερα. Τέλος, συζητούμε πιθανές εφαρμογές υψηλότερου επιπέδου, όπου η μέθοδός μας θα μπορούσε να αποδειχθεί χρήσιμη ως ένα αρχικό βήμα προεργασίας.

Λέξεις κλειδιά: όραση υπολογιστών, εκμάθηση μηχανών, ταξινομητής, ανίχνευση κορυφογραμμών, συμμετρία, χαρακτηριστικά, σύνολο δεδομένων επαλήθευσης, αξιολόγηση, λογιστική παλινδρόμηση, εκμάθηση από πολλά στιγμιότυπα, ακρίβεια, επανάκληση, εκπαίδευση.

Acknowledgements

I became acquainted with the field of Computer Vision through the homonymous undergrad course, taught by professor Petros Maragos, in the National Technical University of Athens. I found very interesting the fact that Computer Vision can be a link between numerous and diverse disciplines, such as biology, physics, mathematics, and computer science, as well as the seemingly infinite boundaries in present and future applications, and that is why I decided to do my dissertation on this domain.

I would like to thank professor Maragos, both for being an inspiring teacher, and for giving me the opportunity to cooperate with Iasonas Kokkinos in the elaboration of this diploma thesis. Professor Kokkinos supervised the biggest part of this work, providing me with useful ideas and advice regarding the subject and the direction of the thesis, while urging me to explore new aspects of the problem at the same time. His aid and guidance have been crucial towards the completion of this work. Except for my teachers, I would also like to thank my good friend and colleague Ioannis Gkioulekas, for his constant support and help, throughout our studies in the NTUA, until now. He has been a devoted friend and a motivating example while trying to achieve my goals.

Finally I want to express my most sincere thanks to all of my close friends. Meeting them and sharing experiences together, has been one of the most invaluable assets for me during my years as a student. Of course, I cannot forget mentioning my family. Their love, support and encouragement throughout my life have been fundamental in accomplishing my goals.

Contents

List of Figures

Chapter 1

Introduction

1.1 Statement of the problem

Despite the role of symmetry in human perception, it still remains a cue that is rarely used in recognition, classification and scene understanding systems. The latter, along with the fact that symmetry is considered a pre-attentive feature that enhances object recognition [8, 22, 44] were our main motivations in trying to develop a system that detects ridges and symmetry axes in natural images.

Specifically, this diploma thesis deals with the problem of automatic ridge and symmetry axis detection in natural images, using an appropriately trained detector, that can decide if a pixel belongs to a symmetry axis or not by using features extracted from the input image, which can be either color or grayscale. Given a new image, the detector extracts the same features and calculates the probability for a pixel to belong on a symmetry axis at some scale and orientation, classifying it accordingly. The novelty in our approach lies in the fact that we use supervised learning to train our detector; although similar approaches have been adopted for boundary detection [12, 20, 29] and corner detection [37] yielding useful results, there seems to be no relevant work that we know of for symmetry and ridge detection.

This formulation differs from previous attempts to tackle the problem, which use mostly geometric methods, and are based for the largest part only on the brightness or boundary cues. On the other hand, our method exploits the additional features of texture and color, which enable the detector to discriminate symmetry in image regions that share a common brightness content or boundary features. Another advantage of our approach is that we can increase the number and nature of features used to perform the training, in order to improve the results, the only limitation being keeping the computational cost tractable. In this thesis, we use a small number of features for the sake of simplicity. We believe that

incorporating a statistical framework in our approach can lead to better results, approximating the decisions a human subject would take, with greater success.

Due to the way we model symmetry and extract features (explained in chapter 4), our detector is mainly focused on detecting ridges of elongated shapes along one dimension, such as human or animal body parts, tree trunks, and rivers or roads in a topographical top view of an area. We use the terms *ridge* and *symmetry axis* interchangeably throughout the whole document and rely on a ground truth dataset of annotated positives to train the detector. This way we maintain the flexibility to specialize the parameters of the detector for applications aiming at detecting ridges of specific objects or structures.

Applications that could exploit our symmetry axis detection system include pose estimation, where the classifier could be trained to detect human parts exhibiting symmetry (limbs, torso, head), and medical imaging, where ridge detection in MRI or PET scans can offer important knowledge concerning pathological conditions. Moreover, our detector could be trained using aerial images, aiming to extract ridges in landscape top views; this step can be of use in topography, e.g. in automatic topographic map extraction.

1.2 Computer vision

This diploma thesis belongs in the research field of *computer vision*. Computer vision aims at developing methods that use two-dimensional images to extract information concerning the part of the 3-dimensional world that is depicted in them [28]. In order to fulfill this goal, research in the field of computer vision is focused mostly on the development of artificial vision techniques, the use of computational models for the study of biological vision, and finally, understanding the sensory and perception processes of the brain that are connected to vision.

Computer vision is an interdisciplinary field born in the decade of 1960 mainly by the contribution of three domains: signal processing, pattern recognition and artificial intelligence. Nowadays it has evolved into an independent and significantly active scientific field that uses techniques from various other domains such as psychology, neurobiology and applied mathematics. Despite the rapid development of Computer vision over the last years, some of the basic problems remain unsolved and even the most sophisticated methods of vision are still far from reaching human vision levels. This indicates that there is a lot of progress still to be made in the years to come, even though that making machines “see” was originally considered to be an undergraduate student summer project.

Finally, considering the crucial role vision has played in the evolution and development of mankind, we can imagine that granting automated systems a similar ability is justified by a plethora of possible applications, the most notable of which

include: mobile robot navigation, industrial inspection and military intelligence, medical image analysis, object recognition, human-computer recognition, and the realistic rendering of synthetic scenes in computer graphics.

1.3 Machine Learning

Machine Learning is a scientific discipline that can be mostly related to artificial intelligence and statistics, and its main purpose is to explore ways and algorithms that grant computer systems the ability to evolve behaviors and make choices based on empirical data, such as data from sensor or databases. This can be accomplished by using a set of observed examples (*training data*) as input to the learning system, and a procedure called *training* so that the learning system can capture underlying statistical laws the data obey.

A crucial trait that is sought after in a learning system is good generalization. The latter represents the ability of a learner to make "intelligent" decisions and provide useful output in new test cases, by using just a small subset of the possible inputs as training data, hence making training a feasible task in terms of time and computational resources.

Learning problems can be divided into two main categories: *supervised* and *unsupervised*:

Supervised learning, or "learning with a teacher", is the task of inferring a function from supervised training data. In this case, the training data consist of a set of *training examples* and each example comprises a *pair* of an input object and a desired output value. The input object is usually a vector in a d-dimensional space, while the ouputs are usually categorized as either *quantitative* or *qualitative*. Qualitative outputs are also referred to as *categorical* or *discrete* variables, as well as factors. This distinction in the output type has led to a naming convention for the prediction tasks: *regression* when we predict quantitative outputs, and *classification* when we predict qualitative outputs [16]. Supervised learning is the type of learning we use in this project.

Unsupervised learning attempts to capture the hidden structure in data that have not been labeled. The usual setting for this problem is that one has a set of N observations of a random p-vector X, having joint density P(X) and the goal is to infer the properties of this probability density without the help of a labeling providing correct answers or a measure of error that needs to be minimized.

Well-known approaches to unsupervised learning include clustering methods like k-means and dimensionality reduction techniques like principal component analysis and independent component analysis.

1.4 Thesis outline

Chapter 2 provides a short review on the previous work on symmetry axis and ridge detection. It also present work that is relative on a machine learning level and that motivated our approach.

Chapter 3 describes the construction of the ground truth dataset used to train of the classifier. It analyzes the steps involved in its construction and presents an interface that includes supervision by a human user.

Chapter 4 presents the features that were used in the training of the ridge detector, as well as the method we used for their collection. We introduce an adaptation of the gradient-based operators that were formulated in [29], which we use to compare the extracted features and detect symmetry.

Chapter 5 provides the theoretical background for the tools which are used for the detector training, namely *logistic regression* and *Multiple Instance Learning*. The details of the training procedure are explained and the different configurations for the problem parameters are explored.

Chapter 6 describes the testing procedure using the training methods presented in chapter 5. The evaluation method in the *precision-recall* curves framework is explained and used to compare the results quantitatively. Logistic regression and multiple instance learning results are also compared to Lindeberg's automatic scale selection technique. In this chapter we also include indicative qualitative detection results.

Chapter 7 summarizes the thesis and presents the conclusions we have reached. In this last chapter we also propose possible practical applications where our work could be useful, and finally, we list improvements and alternative approaches that we intend to explore in the future.

Chapter 2

Symmetry and previous work

This chapter is composed of two parts. In the first part we provide some general information on basic symmetry concepts that are useful in understanding the problem we are trying to tackle. In the second part, we go on making a short review of the previous work on symmetry axis detection and ridge detection. Since there is not a single accepted definition for what a ridge is, we list the various definitions we encountered in the bibliography and the respective algorithms.

2.1 Symmetry

The scientific interpretation views symmetry as the repetition of patterns or self-similarity. This property can be demonstrated in a variety of scientific fields and various forms. Below we list basic types of symmetry, which provide a better understanding of the problem we are trying to tackle.

2.1.1 Symmetry in Euclidean Space

Symmetry in the Euclidean space is formally defined in the following manner: Consider a subset of $\Re^n$, S. A distance preserving mapping (*isometry*) g is a *symmetry* of S, if and only if $g(S) = S$. Additionally, a symmetry g for a set $S \in \Re^n$, is called a *primitive symmetry*, if and only if for any non-trivial decompositions of $g = g_1 g_2$, neither g_1 nor g_2 is a symmetry of S.

For example, given a function $f(x, y)$ in 2D Euclidean space $\Re^2$, which is the Euclidean space of interest in our work, four basic types [9, 11, 50] of symmetries are:

1. *Reflection:* If $f(x, y)$ is a function that is symmetric along axis y, the following relationship holds: $f(x, y) = f(-x, y)$. If f is the image brightness

function for example, the image is symmetric along a specific axis. The reflection (symmetry) axis is the set of points remaining invariant under the reflection.

2. *Rotation:* Rotational symmetry is described by the relationship $f(x, y) = f(\sqrt{x^2 + y^2}\cos(2\pi/n)), \sqrt{x^2 + y^2}\sin(2\pi/n)$, where n is an integer. For this type of symmetry, the point remaining invariant under the rotation is the rotation center.

3. *Translation:* In this case, if $f(x, y)$ is a function exhibiting translational symmetry, $f(x, y) = f(x + \Delta x, y + \Delta y)$. The quantities Δx, Δy belong to $\Re$ and there are no invariant points.

4. *Glide Reflection:* A glide reflection can be expressed as $g = t \odot r$, a translation t followed by reflection r, whose axis of reflection is along the direction of the translation. For example, for a glide reflection, where the translation and reflection are along the horizontal axis, $f(x, y) = f(x + \Delta x, -y)$, for some $\Delta x \in \Re$.

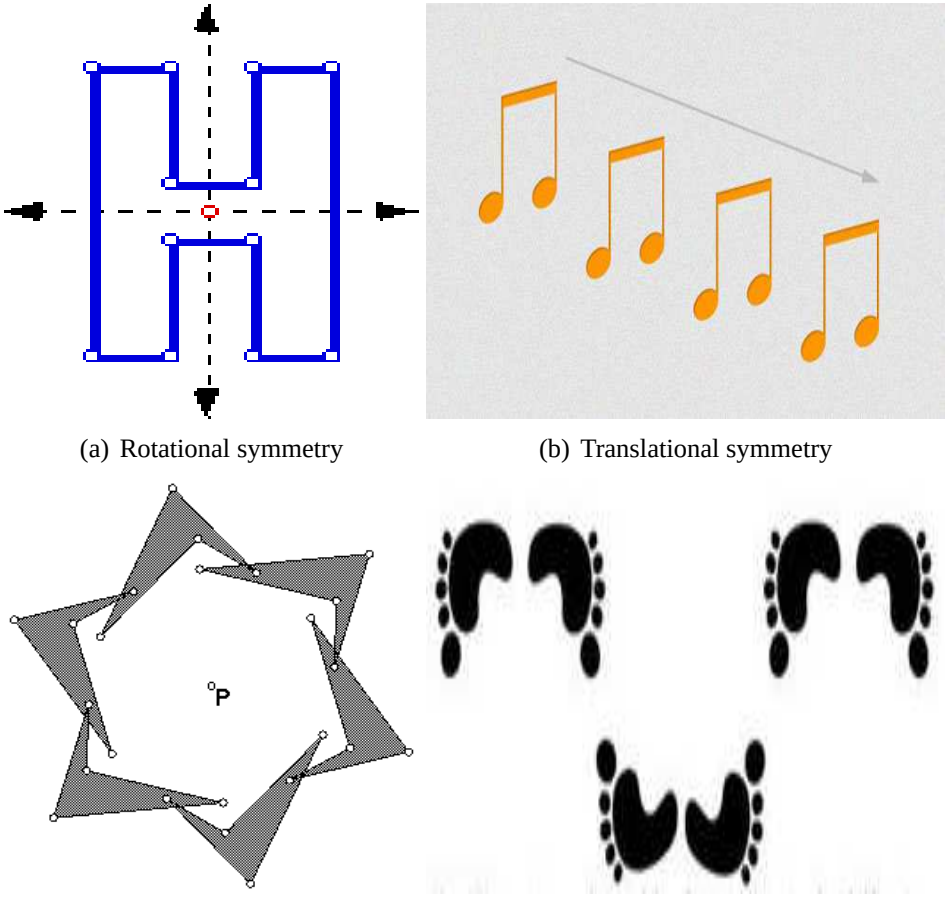
2.1.2 Local and Global Symmetry

Symmetry in natural images can be viewed both as a global and a local phenomenon, depending on the scale at which we are looking for it. For example we consider a simple image containing a foreground object, shown in figure 2.2. For the left object, global symmetry is highlighted; in this case all the object points contribute to the determination of the symmetry along the vertical axis. In the right object of the image, we view symmetry as a local feature and the points composing the illustrated symmetry axes are only supported by a subset of the object. This discrimination can be better clarified by taking a look at Figure 2.2

Global symmetry detection methods are in general much more efficient in run time, usually having linear time complexity [24]; they are, however, sensitive to noise and occlusion. On the other hand, local symmetry detection methods are usually more robust to noise and occlusion, but have high time complexity. One factor that somewhat balances this high time complexity is that most of the times they can also be easily parallelized.

2.1.3 Importance of symmetry

Symmetry is a phenomenon that presents itself in all forms and scales in both natural and man-made environments, extending from biological structures to galaxies,



(a) Rotational symmetry (b) Translational symmetry

(c) Reflection symmetry (d) Glide reflection symmetry

Figure 2.1: Symmetry in 2D Euclidean space.

or even the arts (Figure 2.3). Repeated patterns appear constantly in our surroundings and compose a fundamental element of our perception and understanding of the world. Humans and animals are able to perceive the existence of symmetry in their environment innately, but this is an ability that machine intelligence has not been able to harness yet.

Symmetry has also played a prevalent role in the basic sciences throughout history. Examples vary in the fields of mathematics, geometry, physics, crystallography and biology (Figure 2.4), with some of the most important being the theory of relativity (the discovery of the isometries of Minkowski spacetime under the Poincaré group, the full symmetry group) [31], the double helix structure of DNA (two-fold rotational symmetry) [49] and the discovery of quasi-crystals [39] and their mathematical counterpart, Penrose tiles [34].

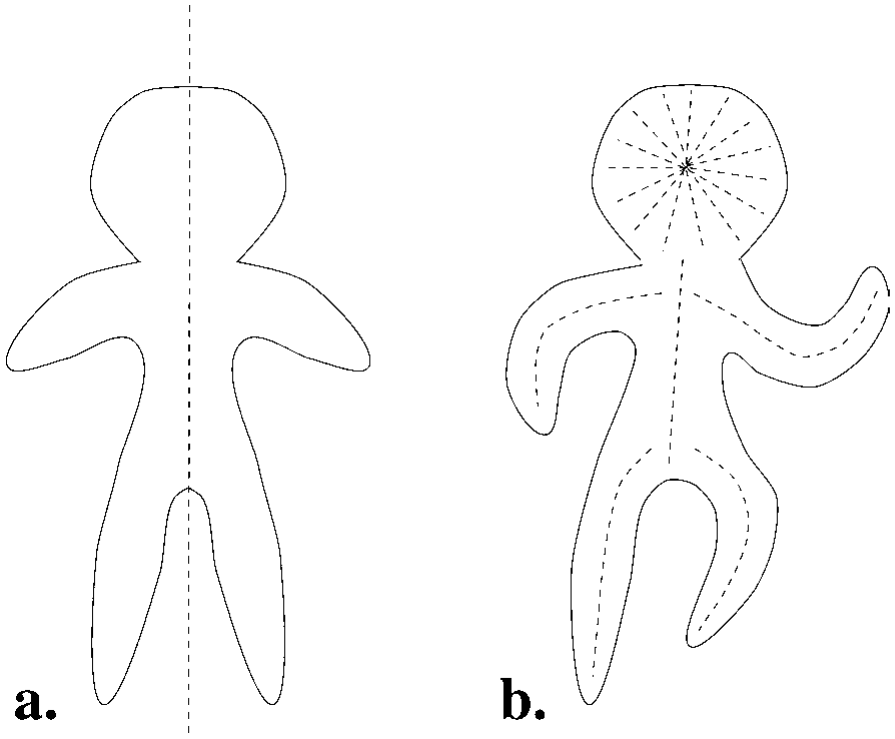


Figure 2.2: Global symmetry (left) compared to local symmetry (right). Figure taken from [24].

2.2 Ridge and symmetry axis detection in 2D images

2.2.1 Binary images

One of the primary works that made use of symmetry axes for shape description was by *Blum* and *Nagel* in 1973 [4]. In this article, Blum used the locus of centers of maximal inscribed circles to define the *symmetric axis transform* of a shape (also called *medial axis* or *skeleton*), previously introduced by the same author in [5]. The function that corresponds the centers of the maximal inscribed circles to their respective radii is called the *radius function*, R. An alternative description of the skeletal axis employs an analogy to grassfire, starting from the boundary of the shape and propagating at unit velocity towards the interior. The quench points of the fire represent the symmetric axis.

In [6] *Brady* and *Asada* use simple geometrical tools to introduce a representation of two-dimensional shapes called *smoothed local symmetries* (SLS),shown in 2.5. The geometry of a local symmetry requires the existence of a pair of points A and B on the shape boundary, such that the following condition holds: The

Figure 2.3: Symmetry in nature and in man-made constructions. In the left column from top to bottom, a snowflake, milky way, the galaxy that contains our solar system, and a libelula. In the right column, from top to bottom, the Taj Mahal in India, a modern painting exhibiting symmetry, and yin-yang.

angle formed by the outward normal vector $\mathbf{n}_A$ at A and the line AB that connects the two points must be equal to the angle that is formed by the respective normal vector $\mathbf{n}_B$ at B and line AB. Of course, this implies that the boundary of the shape

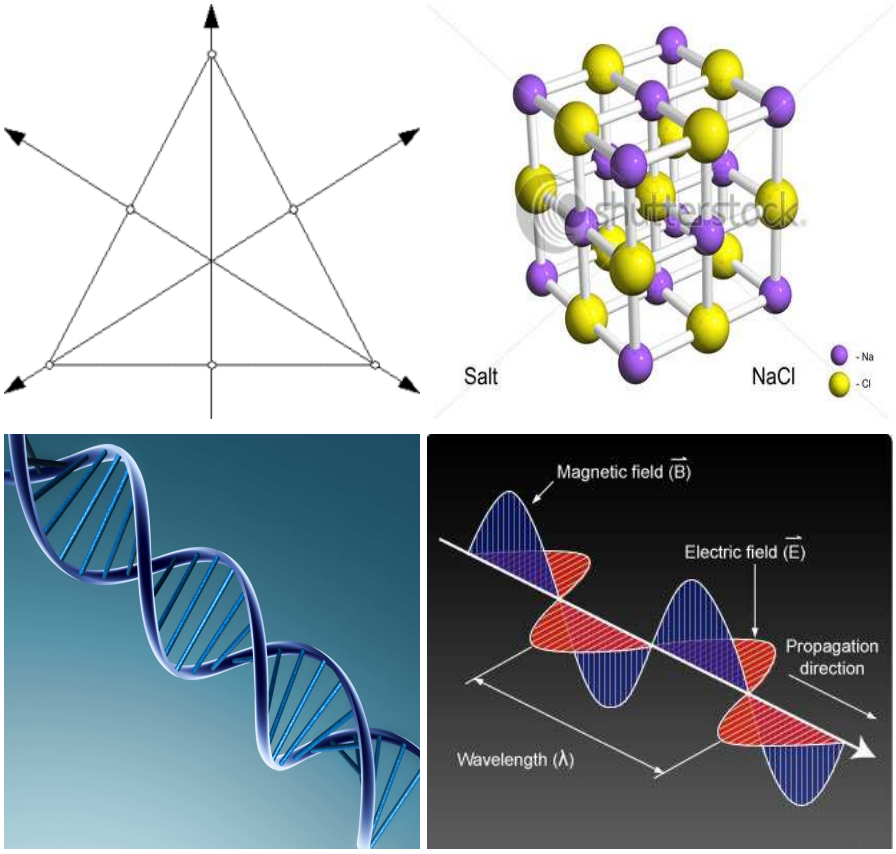


Figure 2.4: Symmetry in sciences.

has been already extracted and that the tangent angle can be computed sufficiently accurately.

As shown in figure 2.5, there may be several points B_i forming a local symmetry with a given point A. The points considered to form the *axes* or *spines* of the shape are the loci of local symmetries forming a smooth curve. Each axis constructed that way, constitutes an alternative way of describing some part of the boundary contour, along with some portion of the region subtended by the axis. This fraction of the shape is called a *cover* of the shape and an hierarchical approach that assigns less importance to axes whose cover is wholly contained in the covers of others is adopted. This also forms an hierarchy among the spines of the shapes according to their rate of locality.

The algorithm for locating the smoothed local symmetries in a binary image, and consequently the medial axes, straightforwardly consists in fixing a bounding contour point A and testing all other points B of the contour for a local symmetry

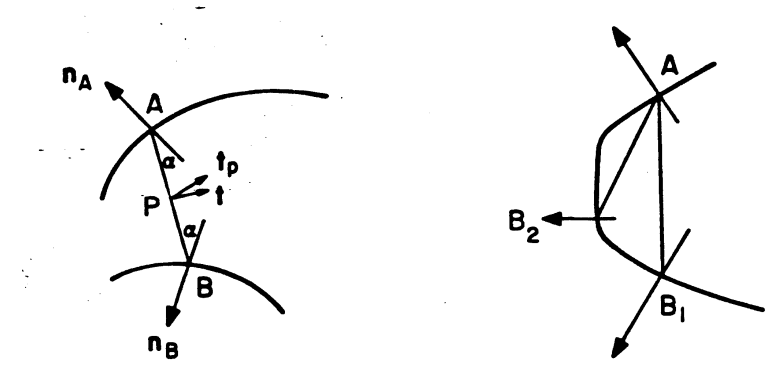


Figure 2.5: Geometry of Smoothed Local Symmetry (left). Point P belongs to a local symmetry axis, with respect to the points A and B. In the right figure, point A forms local symmetries with points B_1 and B_2, illustrating that in general there may be several points B_i forming local symmetries with a given point A (taken from [6]).

with A. This process is repeated for every point in the bounding contour, resulting in $O(n^2)$ complexity. An obvious modification in this algorithm in order to make it more efficient is to test just a subset of the boundary points for local symmetries by sampling the contour. This results in a number of discrete medial points that are an approximation of the continuous medial axis we would get if we used the full contour.

Recent work [45] views the skeleton construction as an optimization problem, trying to prune undesired branches caused by minor boundary deformations, while maintaining a low reconstruction error. This approach results into *canonical skeletons* that are used for shape matching

2.2.2 Grayscale images

Haralick in [17], gives a joint definition for ridges and valleys in grayscale images, using vector analysis concepts. To achieve detection Haralick first translates the notion of ridge and valley to a continuous surface perspective. The concepts that are used for the detection are directional derivatives. More specifically, a two-variable cubic polynomial is first fit to a neighborhood of a pixel, using a coordinate frame whose center runs through the center of the pixel. Ridges and valleys are then found by looking for zero crossings in the first directional derivative, taken in a direction that minimizes (valley) or maximizes (ridge) the second

directional derivative. *Saddle points* are also defined as the points where the conditions for both a ridge *and* valley are satisfied.

Haralick's idea is extended to d dimensional images by *Eberly et al.* in [13], under the term *height condition*. Consider a d-dimensional function L (i.e. for $d = 2$ the intensity function of an image) and a set of indices from 1 to $n - d$, denoted I_{d-n}. If $|\lambda_1| \geq \ldots \geq |\lambda_d|$ are the eigenvalues of $\nabla^2 L$ and $\mathbf{v}_1, \ldots, \mathbf{v}_d$ their corresponding eigenvectors, then a n-dimensional crease is characterized by the equation

$$\forall i \in I_{d-n} : \qquad \nabla L \cdot \mathbf{v}_i = 0 \tag{2.1}$$

Crease is a term used by the authors to collectively refer to both ridges and valleys. Whether the n-dimensional crease is a ridge or valley, depends on the sign of the respective eigenvalue. If $\lambda_i < 0$, then we have a ridge, while if $\lambda_i > 0$, we have a valley. In the same article additional definitions for creases are given, like the *principal direction* definition, which views creases as loci of extrema of principal curvatures along associated lines of curvature in a manner similar to the work of Haralick.

In [25], *López et al* classify the different ridge/valley characterizations according to the area that must be examined for the classification. Ridges/valleys can be characterized as *local* when the classification ox a point **x** as ridge/valley depends on a local test, or *global* when the classification of the point depends on image features arbitrarily far away from **x**. Ridges/valleys that are defined by a local test are called *creases*. There is also the *multilocal* characterization, when the classification of a point **x** depends whether on features of points in a predefined neighborhood or on the particular geometry of the image.

In [35], *Pizer et al.* introduce a new type of medial loci that are called *cores*. Cores are defined as generalized maxima in scale space and their main characteristic is that they convey medial information that is invariant to translation, rotation, and resolution. Given the space of all positions and scales $\Re^n \times \Re^+ \rightarrow \Re$, the authors associate a measure of medial behavior to each point, denoted $M(\mathbf{x}, \sigma) : \Re^n \times \Re^+ \rightarrow \Re$. The medial loci are then extracted by finding the generalized maxima of these measurements.

The functions M satisfying the invariance conditions stated before, are produced by convolving the image with appropriate kernels, based on measures of *boundariness* at each position **x**, scale σ, and orientation **u**, denoted as $B(\mathbf{x}, \sigma, \mathbf{u})$. A simple choice for B can be a measure of luminance change like $B(\mathbf{x}, \sigma, \mathbf{u}) = \mathbf{u} \cdot \sigma \nabla L(\mathbf{x}, \sigma)$.

Functions M are classified as either *central* or *offset*. Central medialness functions attempt to localize object boundaries by averaging spatial information about a point **x** over some region whose average radius is proportional to σ. On the

other hand, offset functions attempt to localize object boundaries by accumulating information of a relatively small scale σ,at neighbors which are at a distance proportional to σ from the test point **x**.

Similar work by *Lindeberg* can be found in [23], where he jointly addresses the problems of edge and ridge detection with automatic scale selection. At each image point (x_0, y_0), a local coordinate system (p, q) is introduced, which is aligned to the eigendirections of the Hessian matrix of the brightness function. The way he formulates a ridge(valley) is in terms of local differential geometric properties of image brightness a connected set of points for which the intensity assumes a local maximum (minimum) in the main eigendirection of the Hessian matrix. In order to formulate a scale selection method for ridge detection, Lindeberg defines the notion of the *scale-space ridge*, as well as a normalized measure of ridge strength $R_{norm}L$, in analogy to other measures of creaseness that have previously been defined for the n-dimensional plane. A scale-space ridge is considered to be the intersection of the ridge surface with the surface defined by $R_{norm}L$ being locally maximal over scales.

Steger in [41] proposes an algorithm that uses an explicit model for lines and their surroundings in the image. Additionally, the algorithm does not aim to simply extract the location of a ridge on the 2D image, but to extract the line position along with the line width, with sub-pixel accuracy. Ridges are located using Gaussian convolution masks and a scale-space analysis. In comparison to other similar approaches though, the existence of an explicit model for the lines and their width extraction, the bias in the extracted line can be predicted analytically and thus be removed. This way, the ridges that are extracted are in semantically meaningful locations in the image. This algorithm is gives as a result the positions of individual ridge pixels in the image. For that reason, after the individual pixels have been extracted, a linking algorithm is used to connect the pixels into lines.

A common framework for edge and line extraction is also derived in the work of *Busch* in [7]. The line model that is used there is a second order polynomial function of the row and column coordinates that is fitted to the grey levels in an image window. Line (ridge) pixels are recognized using the intersecting parabolic function which falls in the direction of the maximal curvature. A pixel is classified as a ridge pixel if the extremum of the parabola falls inside the pixel and the parabola's curvature is sufficiently large. Artifacts and spurious details are separated from salient ridges using hypothesis testing to estimate robust thresholds and additional processing steps like skeletonization and the detection of node and end pixels follow, to improve the results.

Bilateral symmetry and approximate bilateral symmetry is examined in [32]. A measure of symmetry that can be used for grayscale images is proposed. This degree of symmetry is calculated with respect to a certain hyperplane of symmetry and ranges from 0 to 1. Perfect symmetry is achieved when every point on the

one side of the hyperplane has the same intensity value as its symmetric. On the other hand, perfect asymmetry is achieved when every point in one side of the hyperplane has maximum intensity and its symmetric one has the minimum possible intensity value. This algorithm has computational complexity $O(n)$ and is used to measure symmetry only in respect to the principal axes of the object, as *not* all possible planes of a bilateral symmetry are found. The principal axes of the object are extracted automatically too, based on the centroid and the eigenvectors of the covariance matrix of the object.

Chapter 3

Groundtruth construction

In all supervised learning problems, the first step towards the inference of a satisfactory classification function is using a well-constructed ground truth dataset. This dataset consists of a set of inputs and their respective labels, which associate to each input point the appropriate class of the output space.

In the field of Computer Vision, where the ultimate goal is understanding the information that is carried through the optical path, we use annotations by human subjects as ground truth. For example, in the case of the boundary detection learning problem addressed in [29], 300 images with human-marked boundaries from the *Berkeley Segmentation Dataset* (see [30]) were used as ground truth. From the 300 images, 200 and the associated segmentations (5-10 for each image) were used as the training data, and the rest 100 and associated segmentations were used as the test data (we will refer to this specific dataset as the *BSDS300* from now on).

Our goal is to train a binary classifier for ridge detection. We would ideally want a ground truth dataset similar to the BSDS300, comprising images with human-marked ridges. Such a dataset does not exist to our knowledge and its construction would be outside the scope of this thesis. Instead, we combine the information by BSDS300 with a modern skeletonization algorithm based on the work of *Telea et al.* [42, 43, 45] in order to construct fast and automatically a satisfactory ground truth dataset of binary images for training.

We divide the procedure we just described into three main steps:

- Segmentation of input image.
- Skeletonization of every segment separately.
- Union of skeletons for all segments.

Below we describe in more detail each one of these steps.

3.1 Using image segmentations to construct ground truth ridge maps

Image segmentation aims at partitioning an image into a number of segments (sets of pixels) which share some common visual characteristics. Color, intensity and texture are used to quantify the similarity between pixels and ultimately lead to a realistic segmentation which is a simplification of the representation of the initial image. Pixels belonging to the same region have a high degree of similarity between the values of these features, while pixels belonging to adjacent regions have a low degree of similarity.

The problem of image segmentation is closely related to the problem of object recognition, as objects themselves are sets of pixels that usually share common visual characteristics and together form a semantically salient and independent part of the image for the human observer. Natural images are made up of segments such as object parts, animal or human limbs, and others, which are less important, for example large, uniform segments, which usually compose the background (grass, sky or sea). Speaking from a perception standpoint, *Adelson* has referred to this as the distinction between *things* and *stuff* [2].

In order to construct the ground truth for ridges, we use human segmentations taken from BSDS300 such as the one shown in figure 3.1. For each segmentation, we sequentially examine its composing segments and form their skeletons, which will later be used in for the detector training. Segments forming the background do not convey information that can be used in a higher level task like object recognition, so we would like to be able to exclude them from the whole process. To do that automatically we use the *figure-ground labeling data* shown in fig 3.2, which are provided online together with the Berkeley Segmentation Dataset and Benchmark (see also [15]). To decide if a particular segment belongs to the foreground or the background, we take its intersection with both figure contour (f_c) and background contour (b_c). Since the contours encoding this information are adjacent to each other, if the segment has more mutual pixels with the background contour than with the figure contour, then that segment belongs to the background and must be excluded from the skeletonization process. Otherwise, it belongs to a figure and we must take it into account when we create the ground truth.

Applying the skeletonization algorithm sequentially to each one of the segments that compose a specific segmentation, we get several binary images of segment skeletons. In order to form the final skeleton corresponding to the input picture, we have to take the union of all these binary images. An example of the partial skeletons as well as the total skeleton we get for an image segmentation are shown in figure 3.3.

Of course, the above procedure is based on the segmentation of a single human

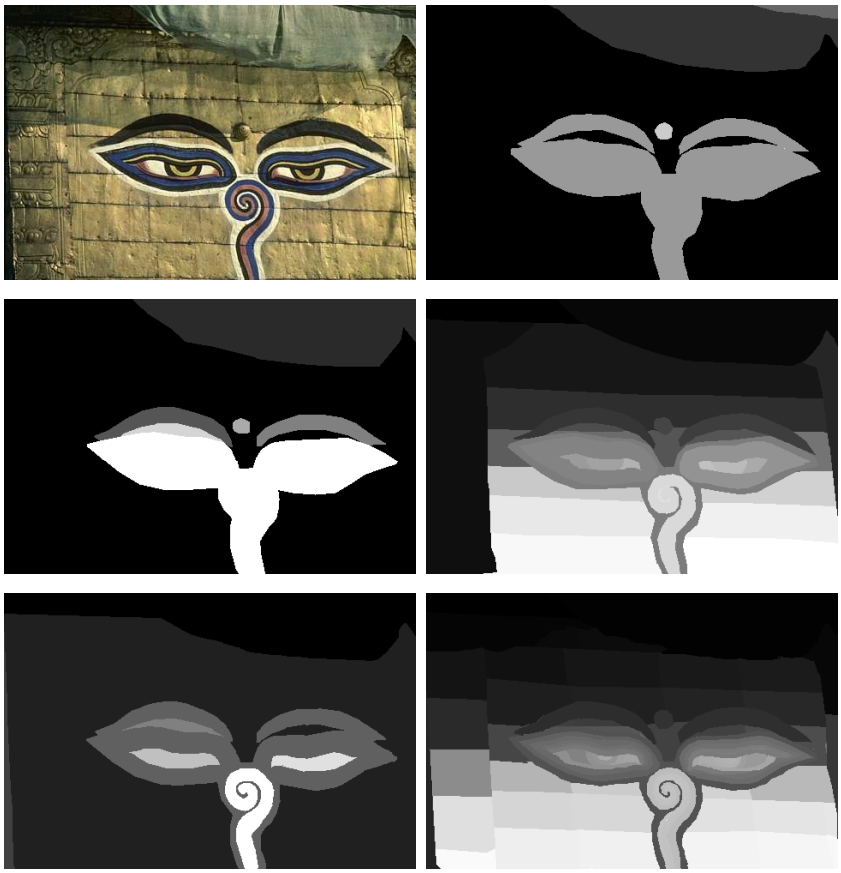
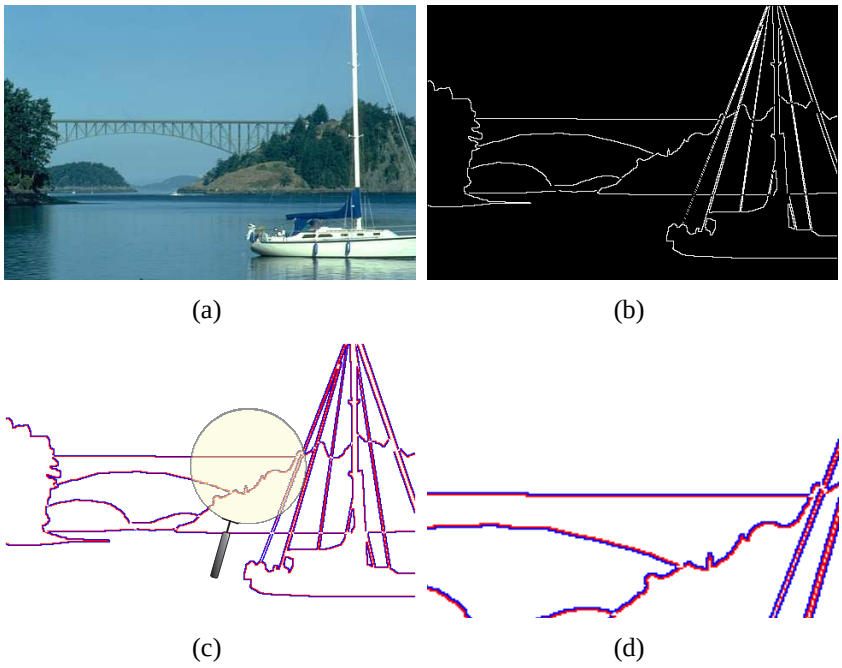


Figure 3.1: Segmentations made by different human subjects for the same image. The varying coloring across the image plane is used to separate segments from one another (a uniformly colored area implies an independent segment).

subject. If we want to take into account different segmentations from other human subjects, we have to repeat the three steps for all the segmentations available for the image and “average” the result in some way. Indeed, what we have done is to repeat the three steps previously described for the segmentations of each image in our training set. This way we get an equal number of different skeletons, whose union we show in figure 3.4.

3.2 Manual screening

In figure 3.5 we see that the resulting skeletons are not always appropriate for training, usually in the case of a segment that is not elongated along some direc-



(a) (b)

(c) (d)

Figure 3.2: Figure-ground labeling encoded as two adjacent contours. (a): Original image. (b): Figure-ground contours, labeled by different human subjects than those who made the segmentations. (c): Figure-ground contours. Figure contours are denoted in red, while their adjacent ground contours are denoted in blue color. (d): Zoomed portion of picture (c).

tion. To deal with this problem, we have created an interface that allows a human user to supervise the creation of the ground truth, by controlling which of the segments will contribute to the creation of the final skeleton. Given a segmentation for the input image, each of the segments is examined separately and shown to the human user, who decides if it should be included in the skeleton or not. This way, we offer the possibility to create a ground truth that approximates the quality of a human-made labeling. In figure 3.6 we show the improvement that can be achieved through human supervision.

3.3 Summary

In this chapter we presented an automated procedure that turns a segmentation problem by a human into a skeleton extraction problem for a given image. This skeleton is used to create the necessary ground truth dataset for the training of the

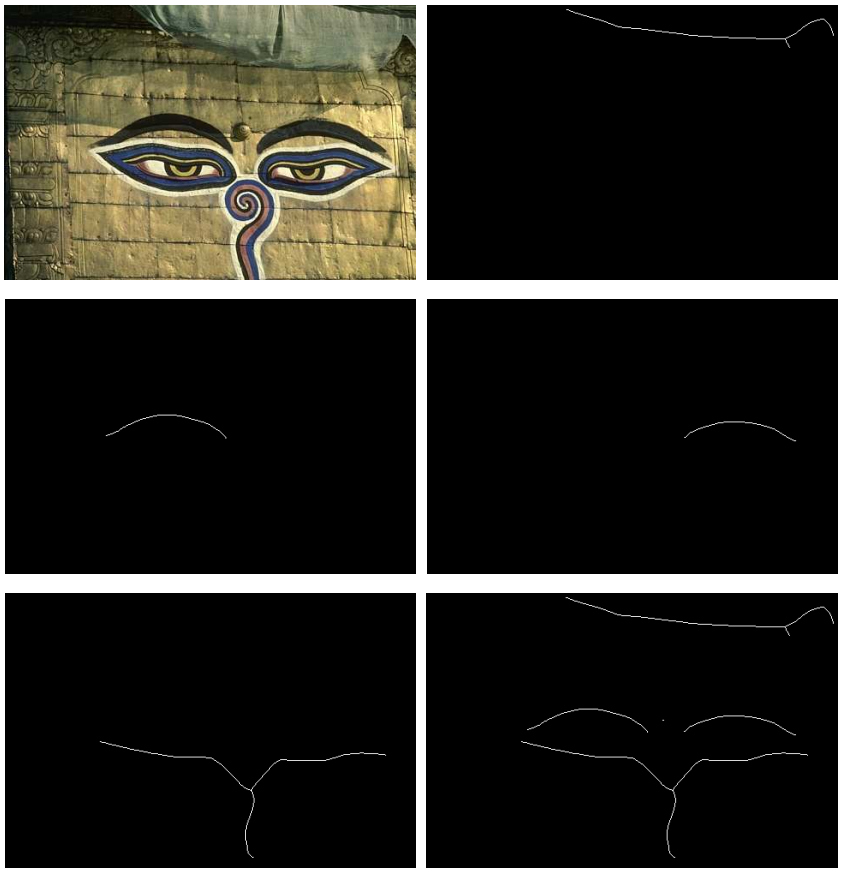


Figure 3.3: Image skeleton composed of the partial skeletons for single segments. This skeleton corresponds to one of the available semgentations for that image.

detector. We went on to analyze the three steps of this procedure, as well as its inherent drawbacks, due to the skeletonization step. Finally, we proposed an alternative approach to the construction of the ground truth, that involves supervision by a human user and can lead to more useful and realistic results.

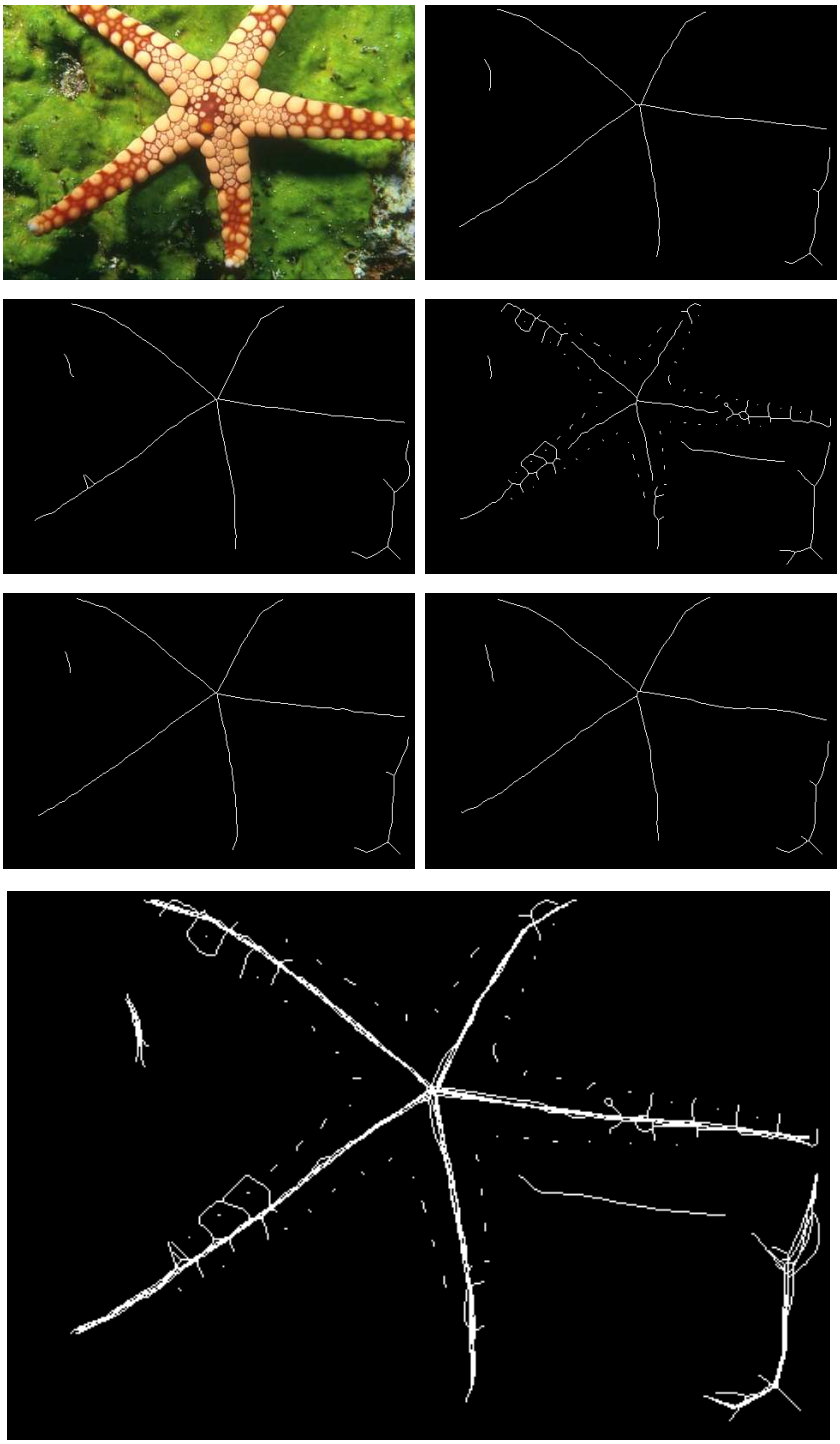


Figure 3.4: Ridge maps derived by using the skeletonization algorithm on different segmentations of the same image. The initial image is illustrated in the top left figure. The large, bottom figure shows the result of the union of the five ridge maps obtained.

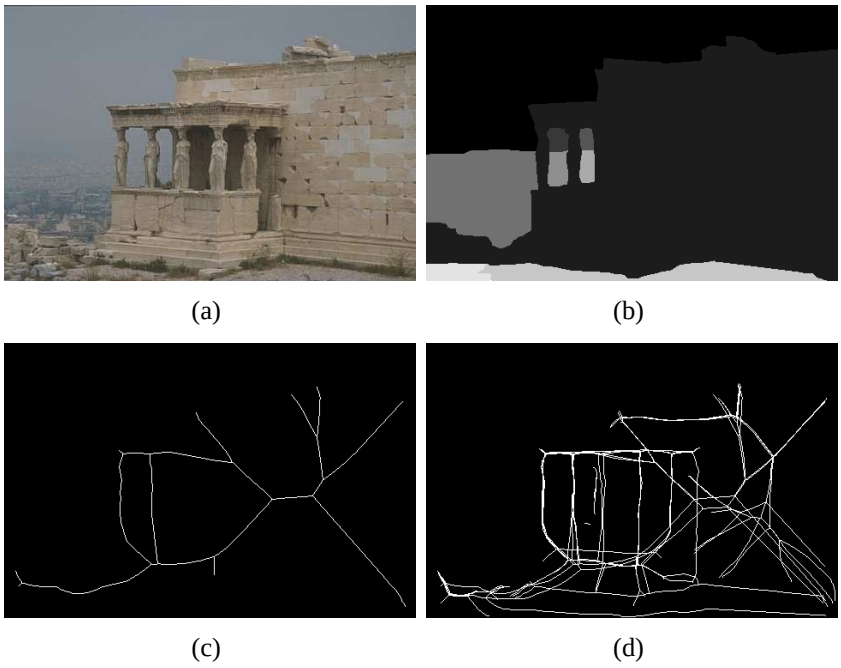


Figure 3.5: Example of the unrealistic result we can sometimes get using the automated skeleton extraction. (a) Original image. (b) Segmentation that contributes negatively to the average skeleton. (c) Ridge map of the large central segment. (d) Union of all ridge maps for the initial image.

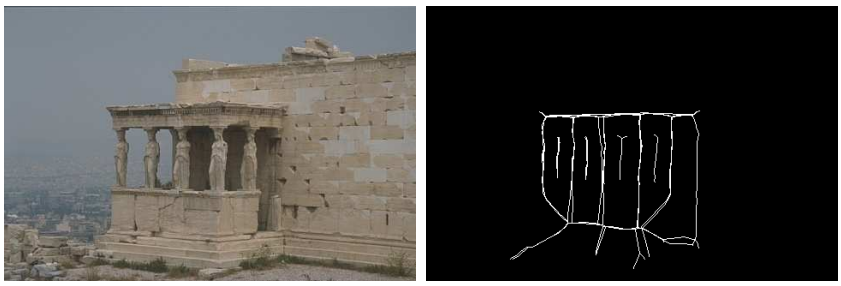


Figure 3.6: Improved union as a result of human user supervision during the selection of the segments contributing to the ridge map.

Chapter 4

Feature extraction

In this chapter we describe the features used to train the ridge detector and the method used to collect them. Our work is inspired mainly by [29], where the authors use features extracted locally from image patches to determine the existence of a boundary in some orientation. Based on the success of the boundary model presented in that paper, we examine if similar local features are exploitable for ridge detection. One significant difference is that in the problem we are addressing we cannot confine our search in a small neighborhood around each point; in figure 4.1 we can see that it is necessary to examine features in multiple scales for each pixel.

4.1 Histogram-based operators and features

We consider a circle of radius r, centered at location (x, y) on the image plane. Drawing two chords along orientation θ and at distance $s = \frac{r}{2}$ from the center divides the circle into three parts as shown in figure 4.2.

Given two of these three parts of the circle, we define the histogram function $H_{D_i,D_j}(x, y, \theta, s)$ that reflects the dissimilarity of the contents of the two parts. Indices i, j are used to denote which two areas of the disk are compared, according to the labeling used in figure 4.2, while quantities θ and s represent the orientation and scale respectively at which we are examining the existence of symmetry. Function H is symmetric with respect to the disk parts, hence $H_{D_i,D_j}(x, y, \theta, s) = H_{D_j,D_i}(x, y, \theta, s)$. Note that the scale at which we search for symmetry *is not* equal to the radius of the circle shown in figure 4.2; the two quantities are connected through the relationship $s = \frac{r}{2}$.

In order to compare the two parts, we make use of color and texture features based on the empirical distribution of pixel values averaged over some neighborhood. There are several approaches which can be used to characterize the differ-

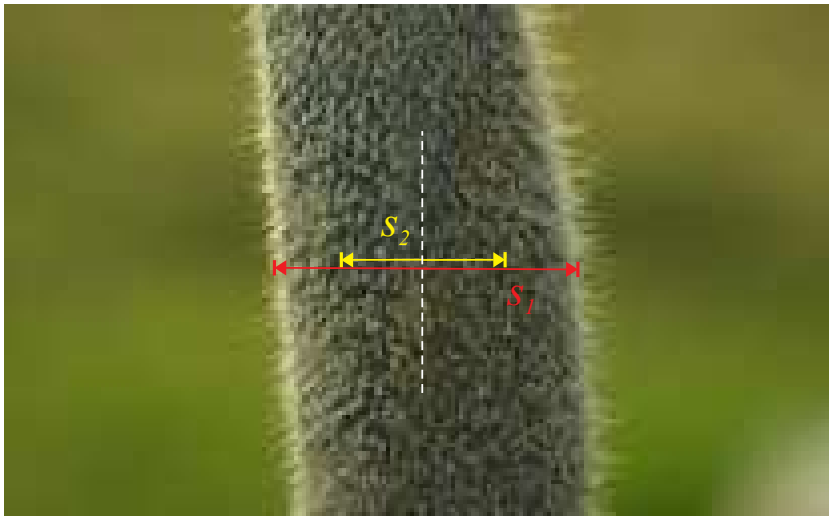


Figure 4.1: Symmetry depends on the bilateral distance of the pixel to the boundary. The same pixel can belong to a symmetry axis with respect to some scale s_1 (red vectors), but not with respect to another scale s_2 (yellow vectors). The vertical symmetry axis is illustrated as a white dashed line.

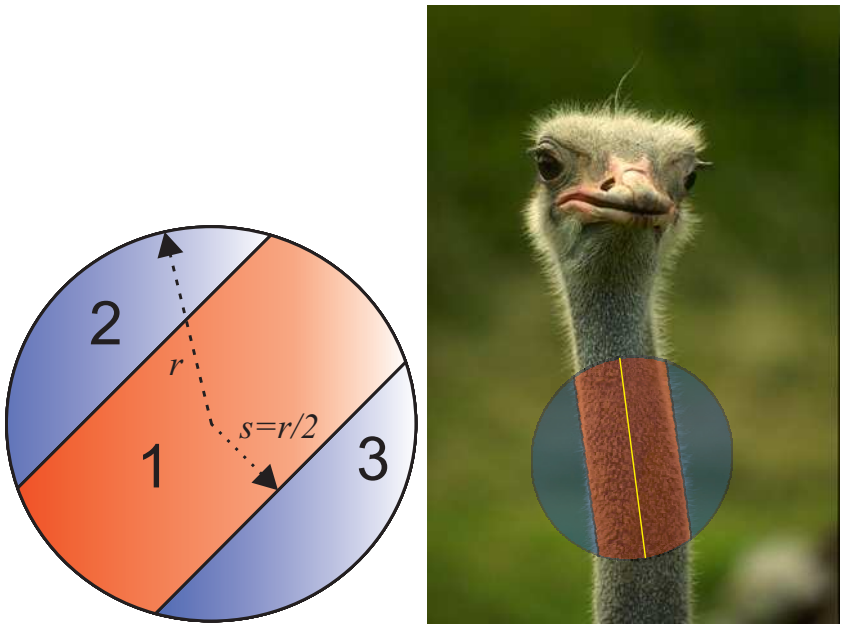
ence between color distributions of two sets of pixels. The *Mallows distance* [27] and the *Earth Mover's distance* (EMD) [21] are common tools for comparing color distributions. Although the latter takes into account the "ground distance" between the points, which can be a desirable property if we are handling data from a color space where nearby points appear perceptually similar, it remains computationally expensive, thus discouraging us from using it. In our attempt to retain a low computational cost, we decided to use the χ^2 difference [38] to compare the histograms. The χ^2 difference measure is described by the following equation, where g and h denote the histograms that are compared and k is the bin index.

$$\chi^2(g,h) = \frac{1}{2}\sum_k \frac{(g(k)-h(k))^2}{g(k)+h(k)} \tag{4.1}$$

Using the notation we introduced for the similarity function between features of different disk parts (fig. 4.2), we have:

$$H_{D_i,D_j}(x,y,\theta,r) = \frac{1}{2}\sum_k \frac{(D_i(k)-D_j(k))^2}{D_i(k)+D_j(k)} \tag{4.2}$$

The computation of the histograms and their comparison is executed for eight orientations in the interval $[0,\pi)$ and ten circle radii. These are set from 0.02 to 0.2 as



(a) (b)

Figure 4.2: (a) Disk used to collect the features. The disk is divided into three parts, whose brightness, color and texture contents are compared. The radius of the disk is denoted as r, while the scale at which the symmetry is detected, is denoted as s. The center of the disk is the point associated with the collected features. (b) Example of symmetry detection using the disk. Feature contents of the middle part(1) show high dissimilarity to the contents of the "left"(2) and "right"(3) parts. Thus, the symmetry response in the vertical diameter, illustrated in yellow, is high.

a percentage of the image diagonal for all three pairs of circle regions; that means we calculate quantities $H_{D_1,D_2}(x, y, \theta, r)$, $H_{D_1,D_3}(x, y, \theta, r)$ and $H_{D_2,D_3}(x, y, \theta, r)$.

The color space we use is the CIELAB color space. CIELAB consists of three channels, the *L** channel for brightness, and channels *a** and *b** for color. For the brightness gradient we compute histograms of the binned L* values. The total color gradient is formed by channels *a** and *b**, so the color value for each pixel lies in a 2D space. According to [33], the *a** and *b** channels correspond to the perceptually orthogonal red-green and yellow-blue color opponents found in the human visual system. This fact motivates the calculation and use of the *a** and *b** gradients as separate features for training. Another option is replacing the joint color gradient CG^{ab} with the sum of the partial color gradients, $CG^{a+b} = CG^{a} + CG^{b}$, whose calculation is much easier; this is again based on the same fact.

In the end of the chapter we include images showing the values of the histogram differences between pairs of circle regions for the brightness and color channels.

In the same spirit as for the brightness and color operators we described in the previous section, we formulate an operator that measures the texture dissimilarity between two areas of the circle disk, at scale s and orientation θ. For the texture feature, we first convolve the image with a bank of filters of various orientations and spatial frequences, as there are several previous works [14, 36], which indicate that the use of such a preprocessing step exhibits good discrimination results. Figure 4.3 shows the filterbank that is used for the texture processing. It contains six even/odd quadrature pairs of elongated, oriented filters, as well as a center-surround filter, the even symmetric filters being Gaussian second derivatives, and the odd-symmetric filters, their corresponding Hilbert transform. Finally, the center-surround filter is a difference of Gaussians.

The 13 filter responses define a 13-dimensional feature space, and each pixel in the two disk regions associated with this vector of responses, represents a point in this space. Following the *texton* approach, as in [29], we cluster the 13-dimensional response vectors using k-means (we used $k = 64$ for our experiments). The cluster centers are the *textons*, and we can see examples of what they look like (for $k = 64$) in figure 4.3. After the textons have been identified, each pixel is assigned to nearest texton, according to its filter response vector. Now texture dissimilarities can be computed by comparing the histograms of texton labels in the two regions of the disk, with the χ^2 difference operator. Figures 4.15-4.17 show some examples of texture gradient features in various scales and orientations.

4.1.1 Boundary Validation Feature

So far we have discussed features which are extracted from an image and used to detect symmetry. Local symmetry in the form of a ridge is also closely related to the existence of boundaries at equal (or approximately equal) distances, on both sides of the symmetry axis. For that reason we associate each pixel to a boundary map response at scale s and orientation θ, and create this way a new symmetry feature.

Specifically, we first extract the boundary map out of the initial image, using the boundary detector formulated in [3]. After that, we filter the boundary image with the masks shown in Fig. 4.4. Each one of these masks is nonzero only in two bilateral areas, centered at distance $s = \frac{r}{2}$ (where r is the radius of the circular disk), and along angle θ. We denote these two nonzero areas as $M_L^{s,\theta}$ and $M_R^{s,\theta}$, where L, R stand for $Left$ and $Right$ and s, θ imply the scale and orientation respectively.

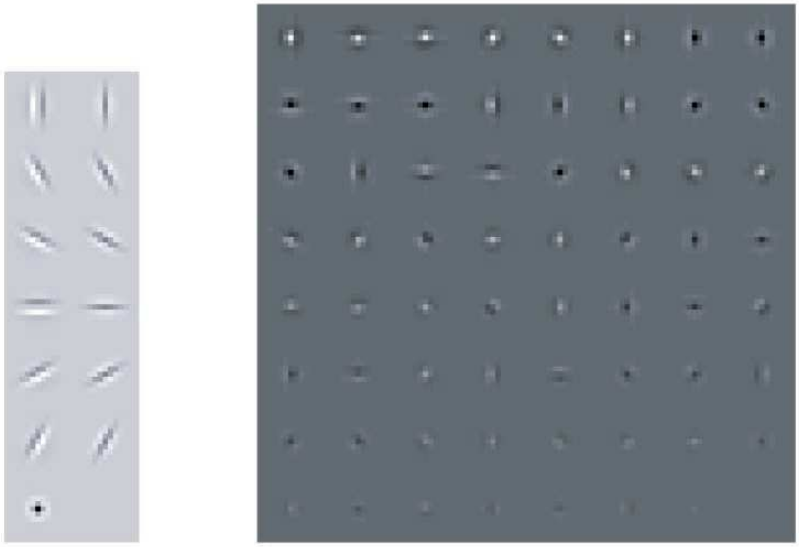


Figure 4.3: The filterbank used to compute textons, composed of 13 filters(left) and an example of computed textons (right). The images are taken from [29].

If there is a high boundary response in both nonzero areas of the mask, then this pixel is likely to belong to a symmetry axis. On the other hand, if the response is low in any of the areas, then the probability of symmetry on the pixel is weakened. The value assigned to each pixel due to the existence of bilateral boundary at scale s and orientation θ is the product of the maximum boundary values found in the nonzero areas. If $BF(x, y, \theta, s)$ is the boundary feature at scale s and orientation θ for pixel (x, y), and BR is the output of the boundary detector, we have:

$$BF(x, y, \theta, s) = \max_{BR \subseteq M_L^{s,\theta}}(BR) \cdot \max_{BR \subseteq M_R^{s,\theta}}(BR) \tag{4.3}$$

This process is repeated for all scales and orientations. Some of the results are shown in figures 4.18-4.20.

4.2 Feature vector combinations

The process described in the previous section results in a 13-dimensional vector of features for each pixel at a specific scale and orientation. That is, three L^* channel histogram-difference features, one for every pair of disk areas ($H_{i,j}^L$), nine more a^*, b^* and texture gradient features ($H_{i,j}^a, H_{i,j}^b, H_{i,j}^t$ respectively), plus the

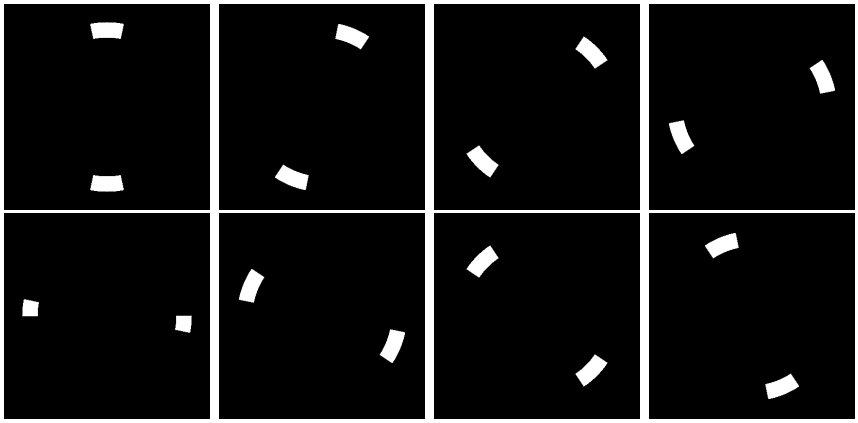


Figure 4.4: Masks used to detect boundary response at distance s from the pixel for all eight orientations. Note that in order to validate the existence of symmetry along angle θ, we examine the existence of boundary at angle $\theta + \frac{\pi}{2}$ (angles are counted clockwise).

boundary validation feature. This 13-dimensional vector of features is extracted at 8 orientations and 10 scales. In the case of grayscale images, we omit the features corresponding to the *a** and *b** color channels, which results in a 7-dimensional feature vector. Also, if we use the sum of the color channel histogram differences H^{a+b} as a single feature we get a 10-dimensional feature vector. To sum up, we consider the following three feature configurations for training:

- Brightness, color, texture and boundary features. Color channel histogram differences are treated as separate features (we will refer to this as the *color* configuration).
- Brightness, color, texture and boundary features. The sum of the color channel histogram differences is treated as a single feature (we will refer to this as the *fullcolor* configuration).
- Brightness, texture and boundary features (we will refer to this as the *gray* configuration).

The optimal way to combine these features in order to detect ridges in new inputs is learned via the learning methods discussed in chapter 5. Intuitively, we believe that points lying on a symmetry axis at orientation θ and scale s exhibit high dissimilarity in the collected features in comparison to their surroundings. That means that for a pixel at position (x, y), $H_{D_1,D_3}(x, y, \theta, r)$ and $H_{D_2,D_3}(x, y, \theta, r)$ have high values. If the bilateral regions of the pixel (areas 1 and 2 of the disk in

fig. 4.2) have similar feature content, we expect that classification becomes easier. This is not always the case though as we can see in figure 4.5, where these two cases are illustrated.

(a) (b)

Figure 4.5: Color and texture features in the two lateral sides of the disk (blue color parts of the disk) can be similar (a) or very different (b). Detection of the symmetry axis is easier in the first case.

4.3 Summary

In this chapter we discussed the features we use for the detector training and the way we extract them. We also introduced a histogram-based operator that uses the χ^2 difference to measure dissimilarity between brightness, texture and color content of two image areas, and added an extra training feature based on the image boundary. Finally, we listed three different feature combinations that will be used for the training of the detector in chapter 5.

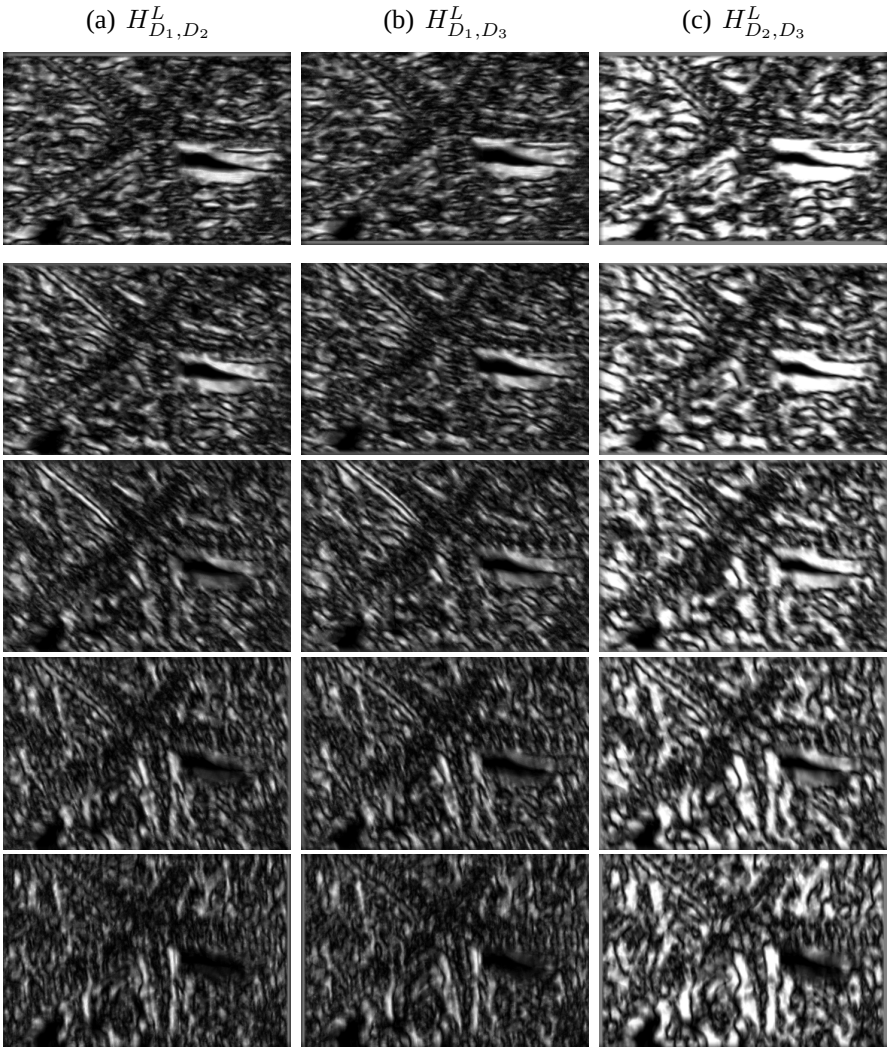


Figure 4.6: Brightness features, collected at the smallest scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

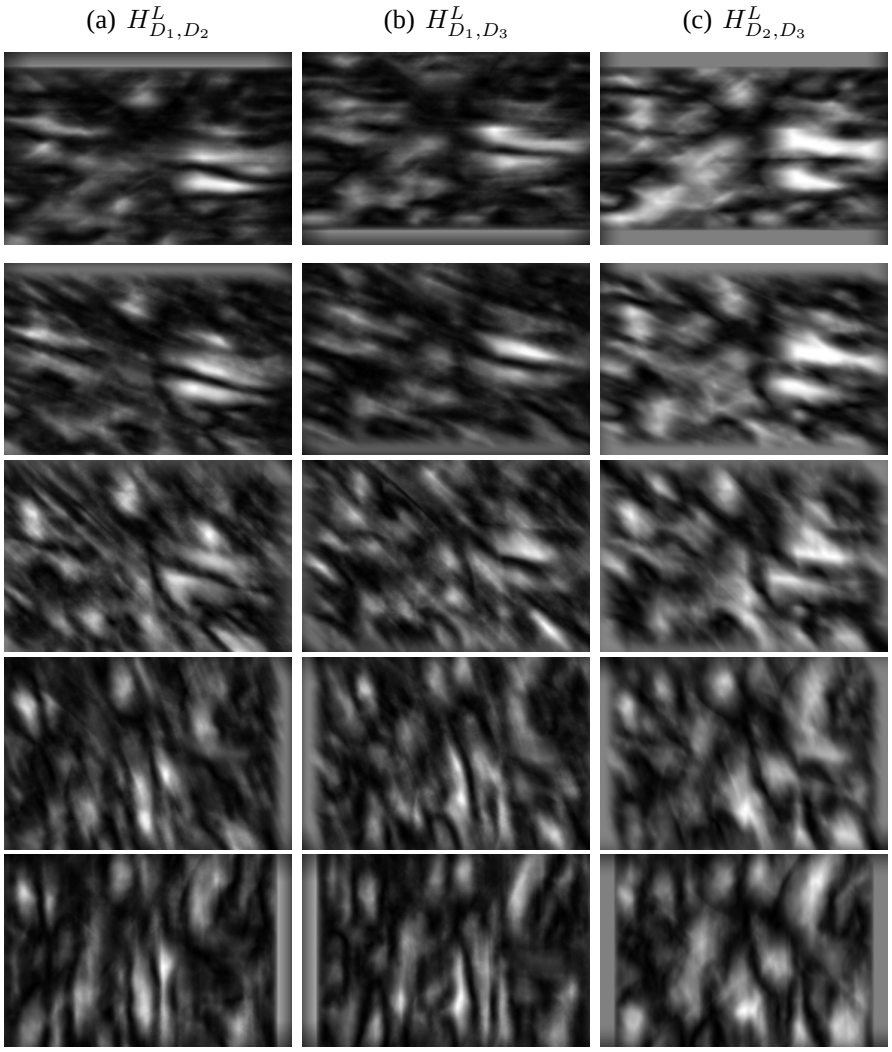


Figure 4.7: Brightness features, collected at a middle scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

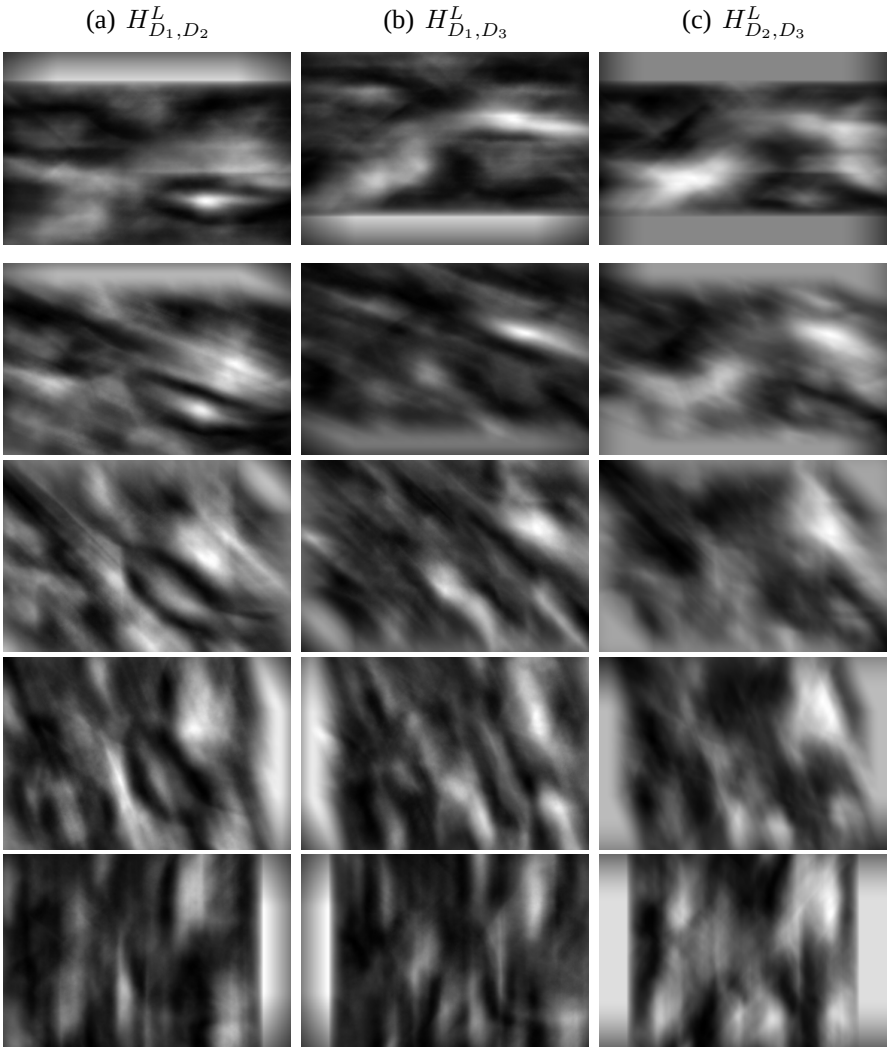


Figure 4.8: Brightness features, collected at a large scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

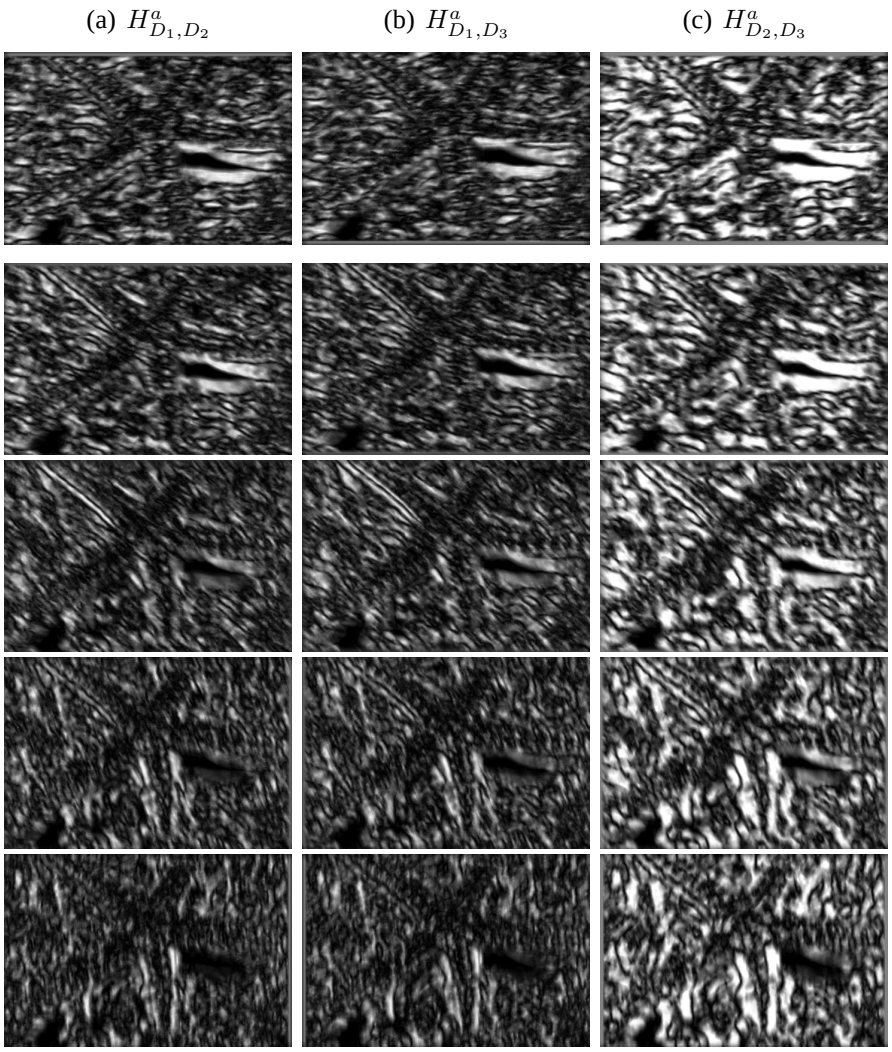


Figure 4.9: Color channel *a** gradient features, collected at the smallest scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

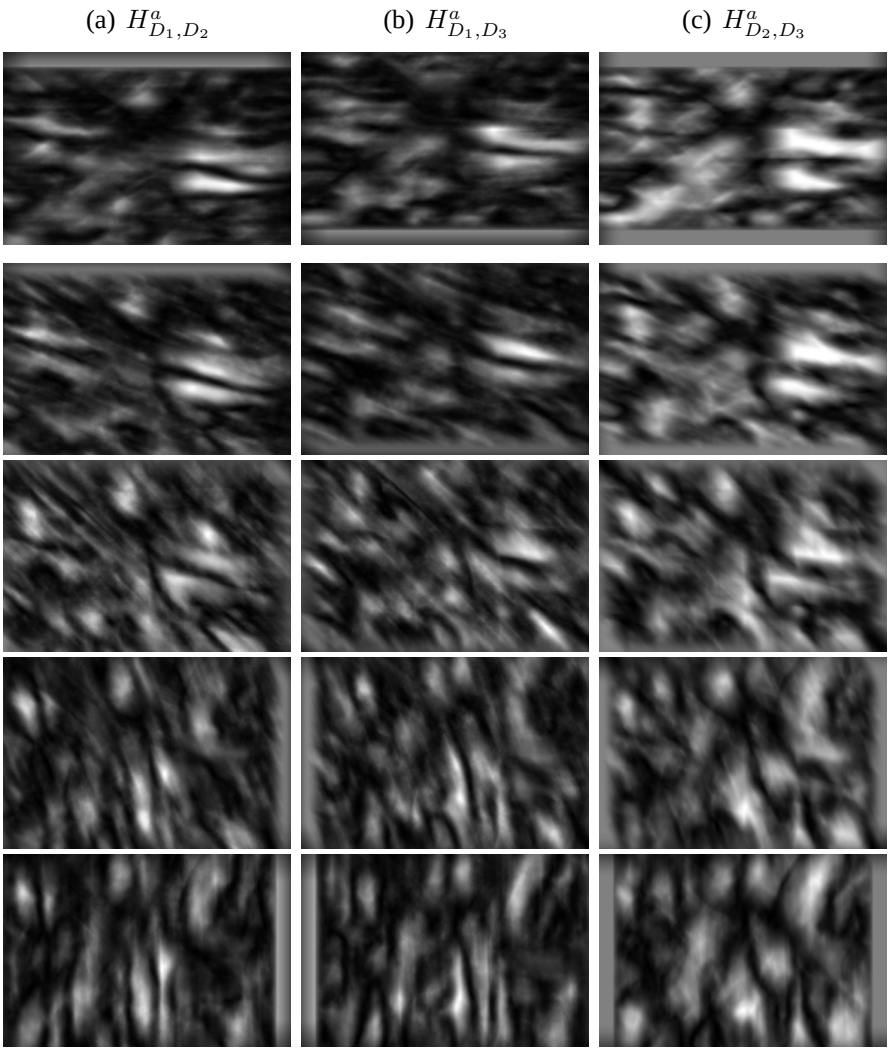


Figure 4.10: Color channel *a** gradient features, collected at a middle scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

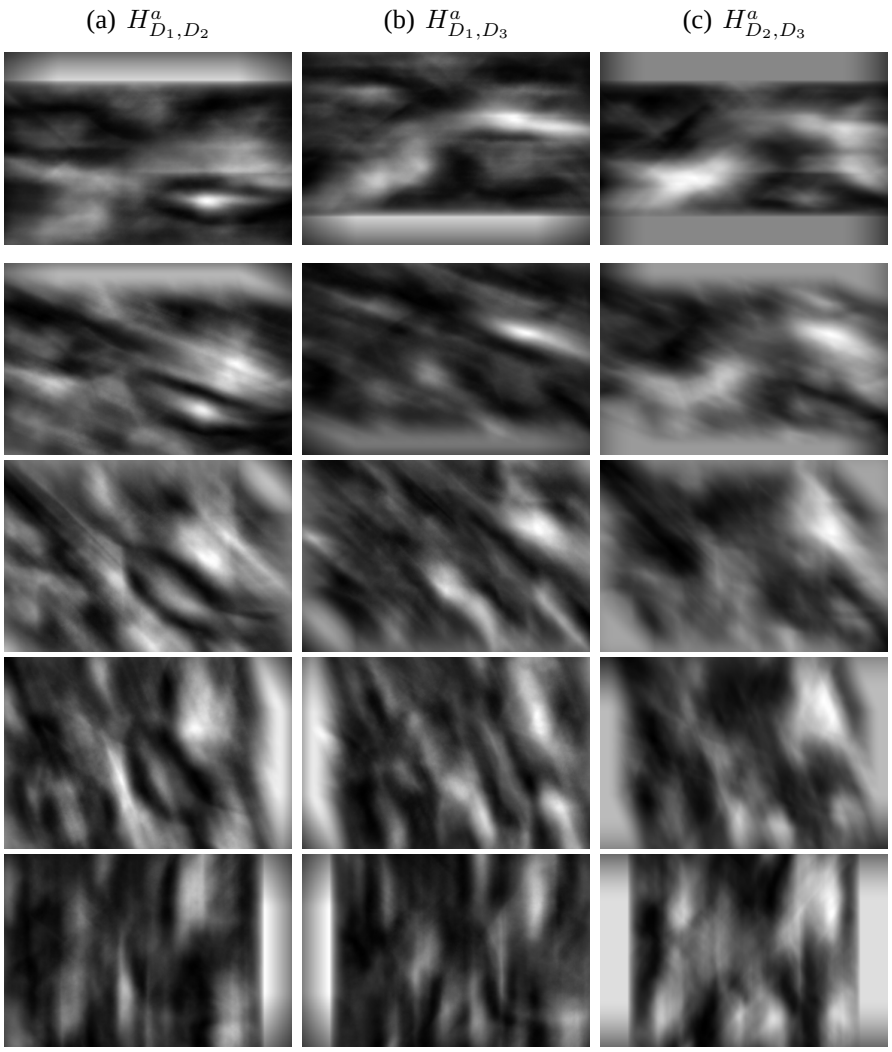


Figure 4.11: Color channel *a** gradient features, collected at a large scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

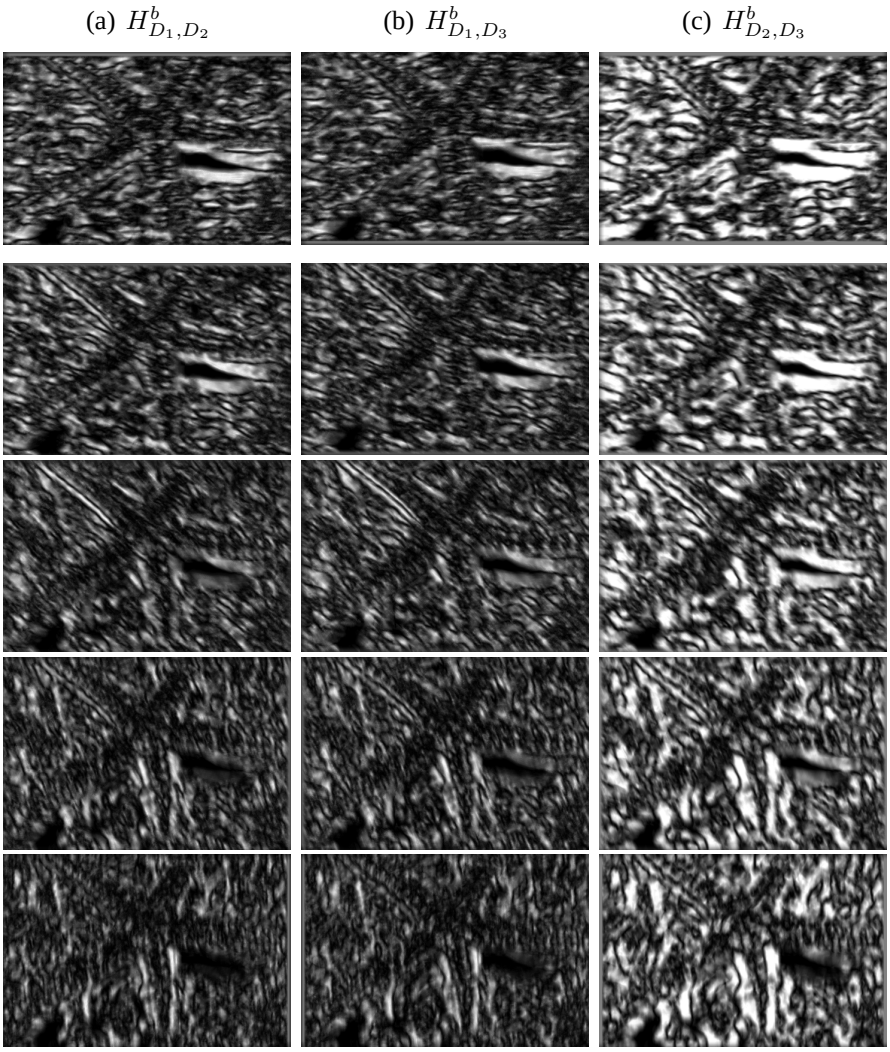


Figure 4.12: Color channel *b** gradient features, collected at the smallest scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

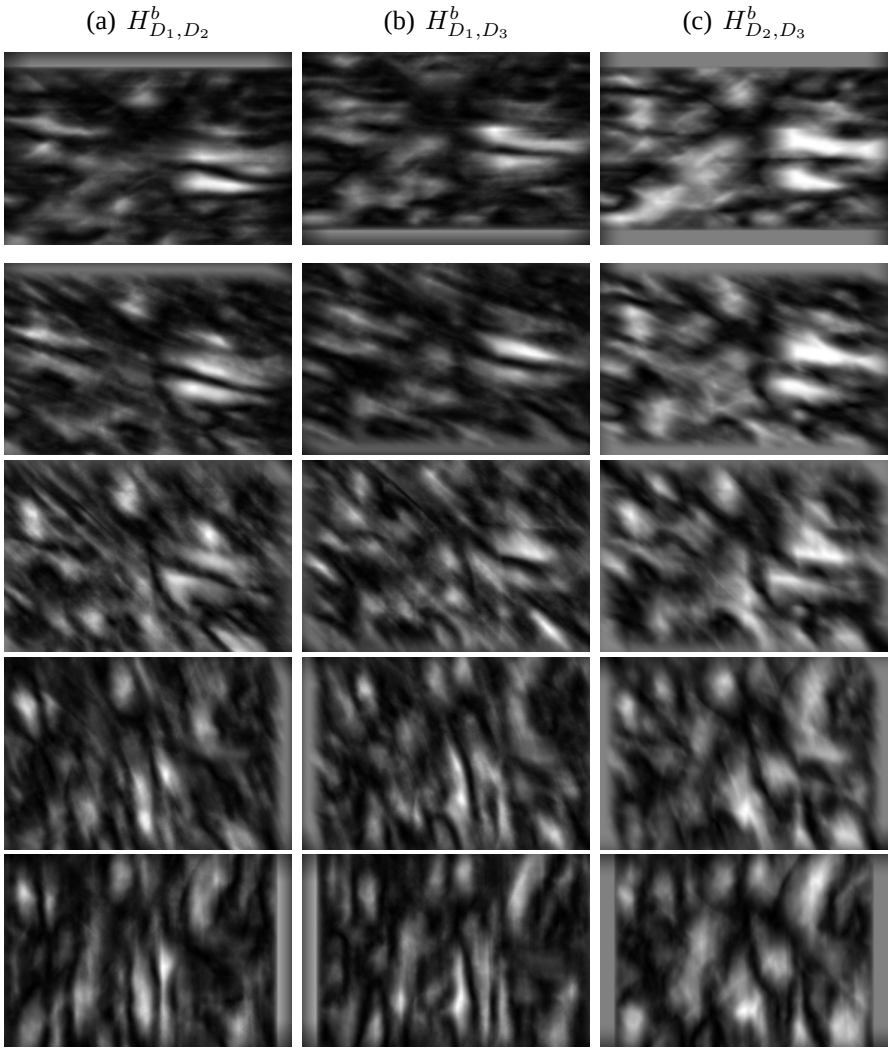


Figure 4.13: Color channel *b** gradient features, collected at a middle scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

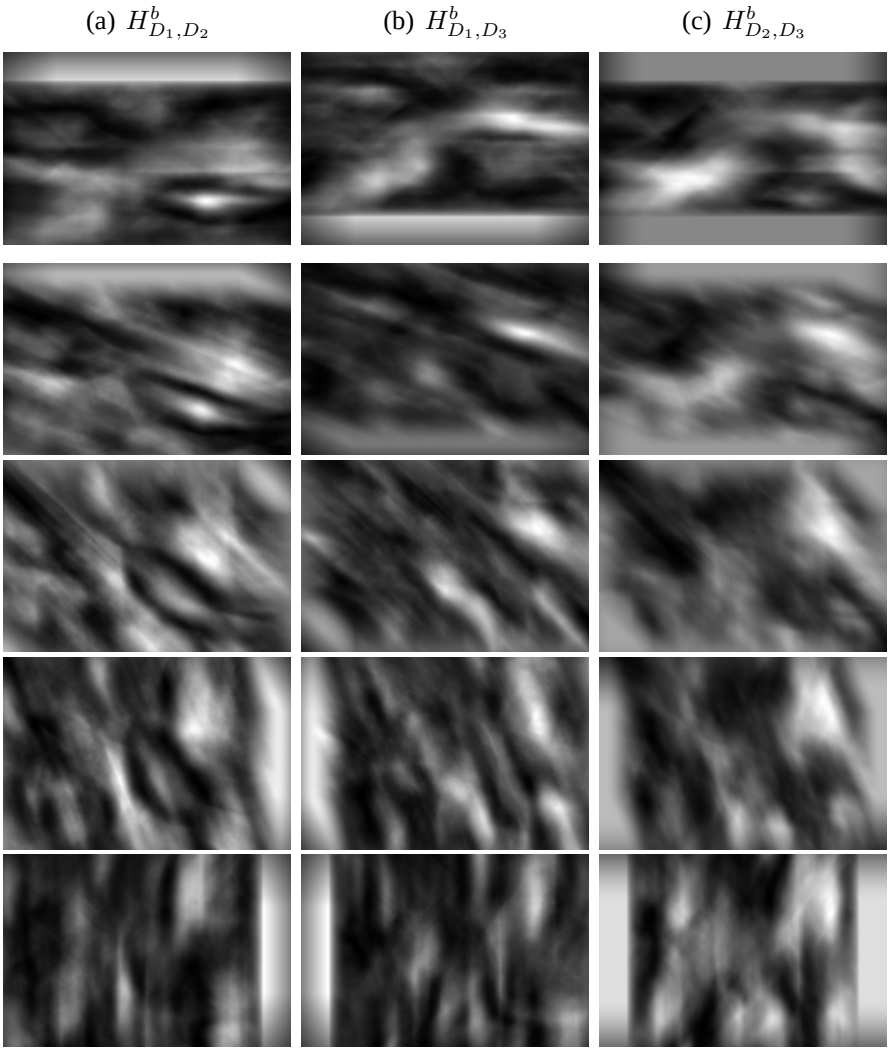


Figure 4.14: Color channel *b** gradient features, collected at a large scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

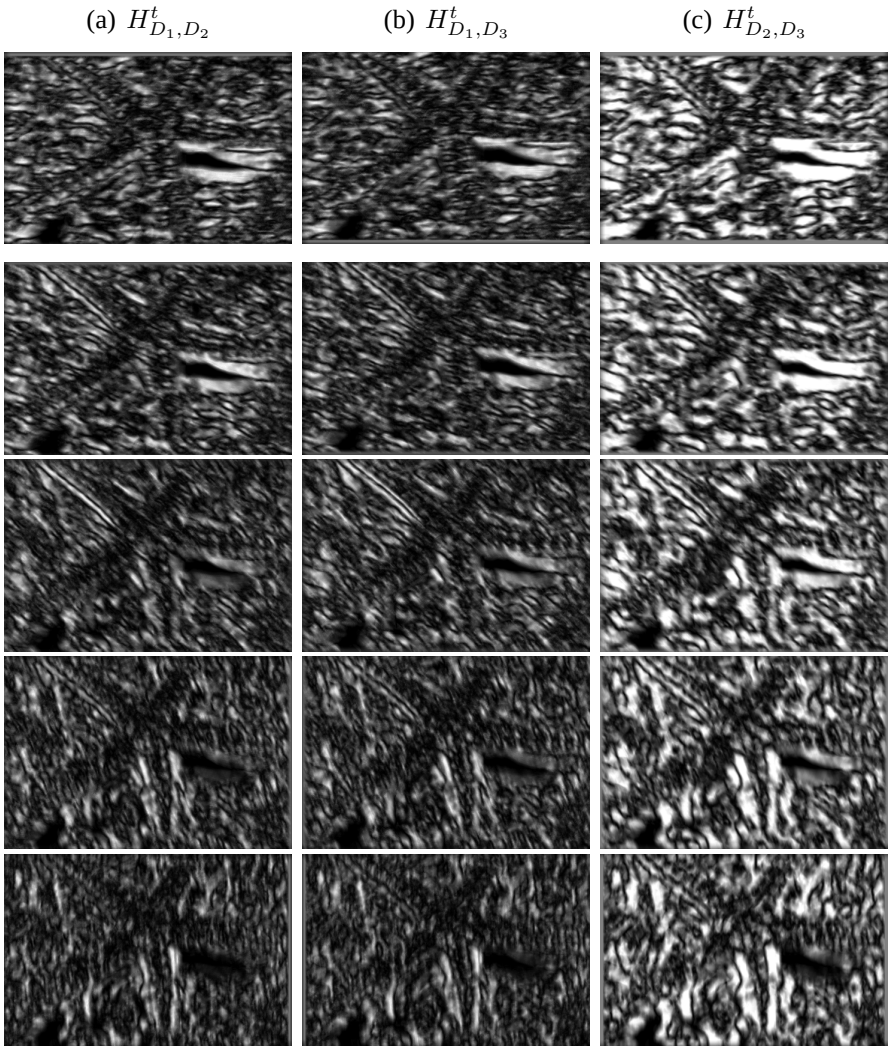


Figure 4.15: Texture channel gradient features, collected at the smallest scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

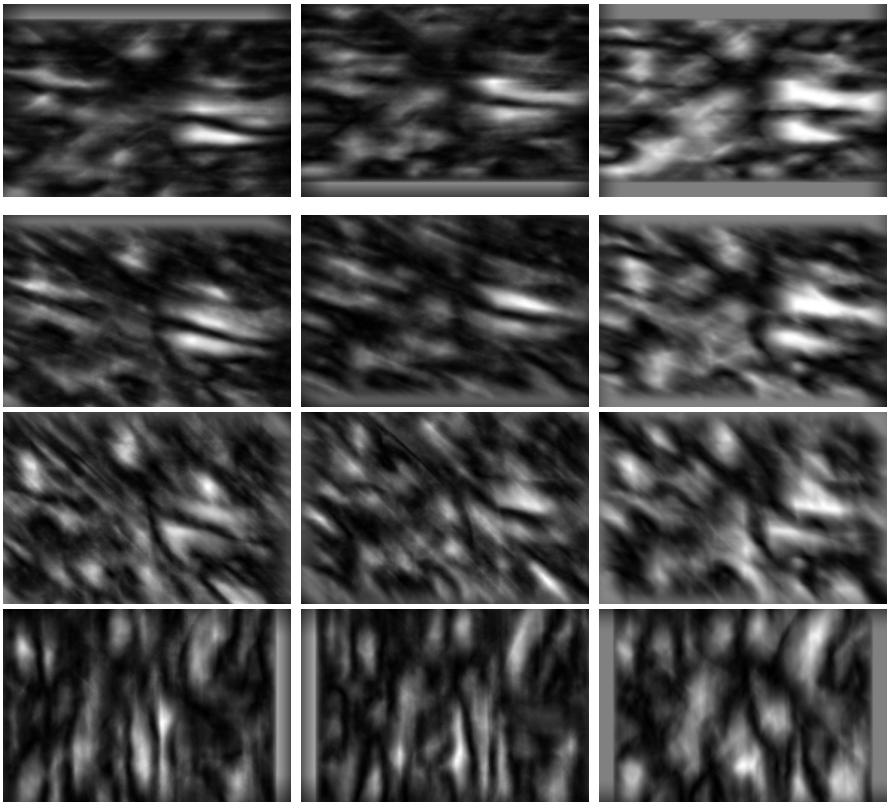


Figure 4.16: Texture channel gradient features, collected at a middle scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).

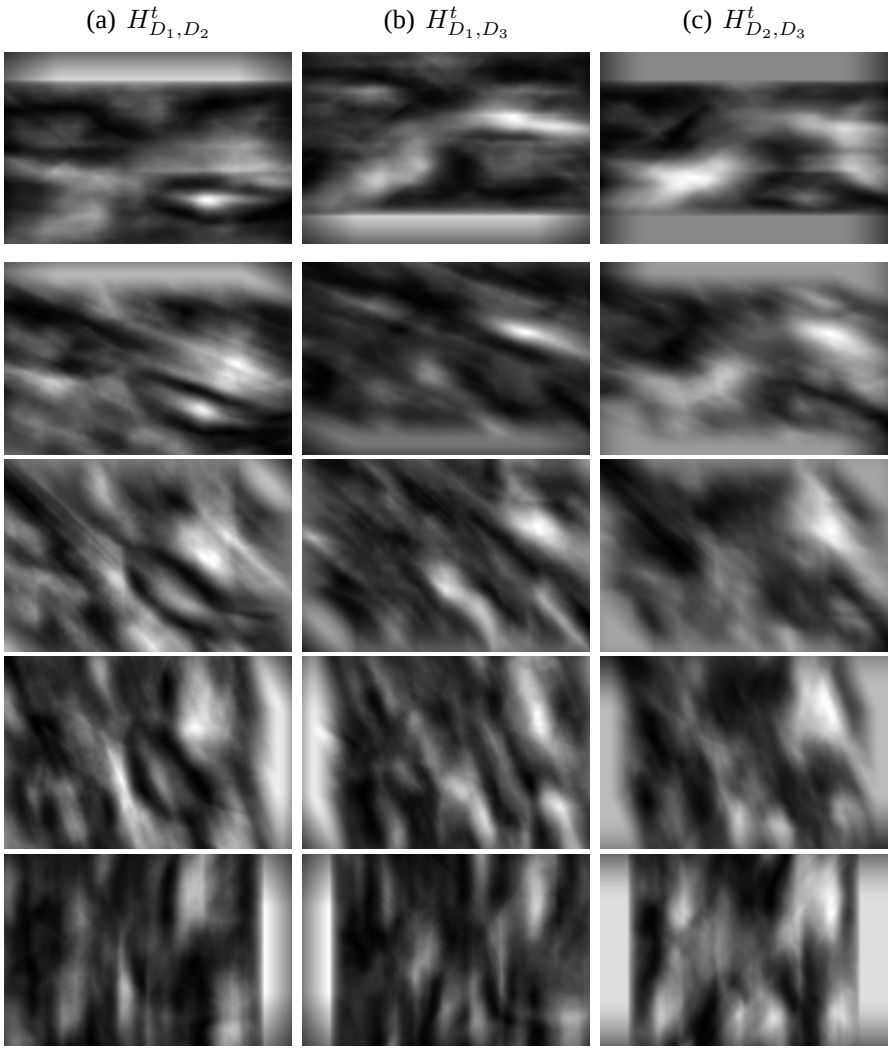
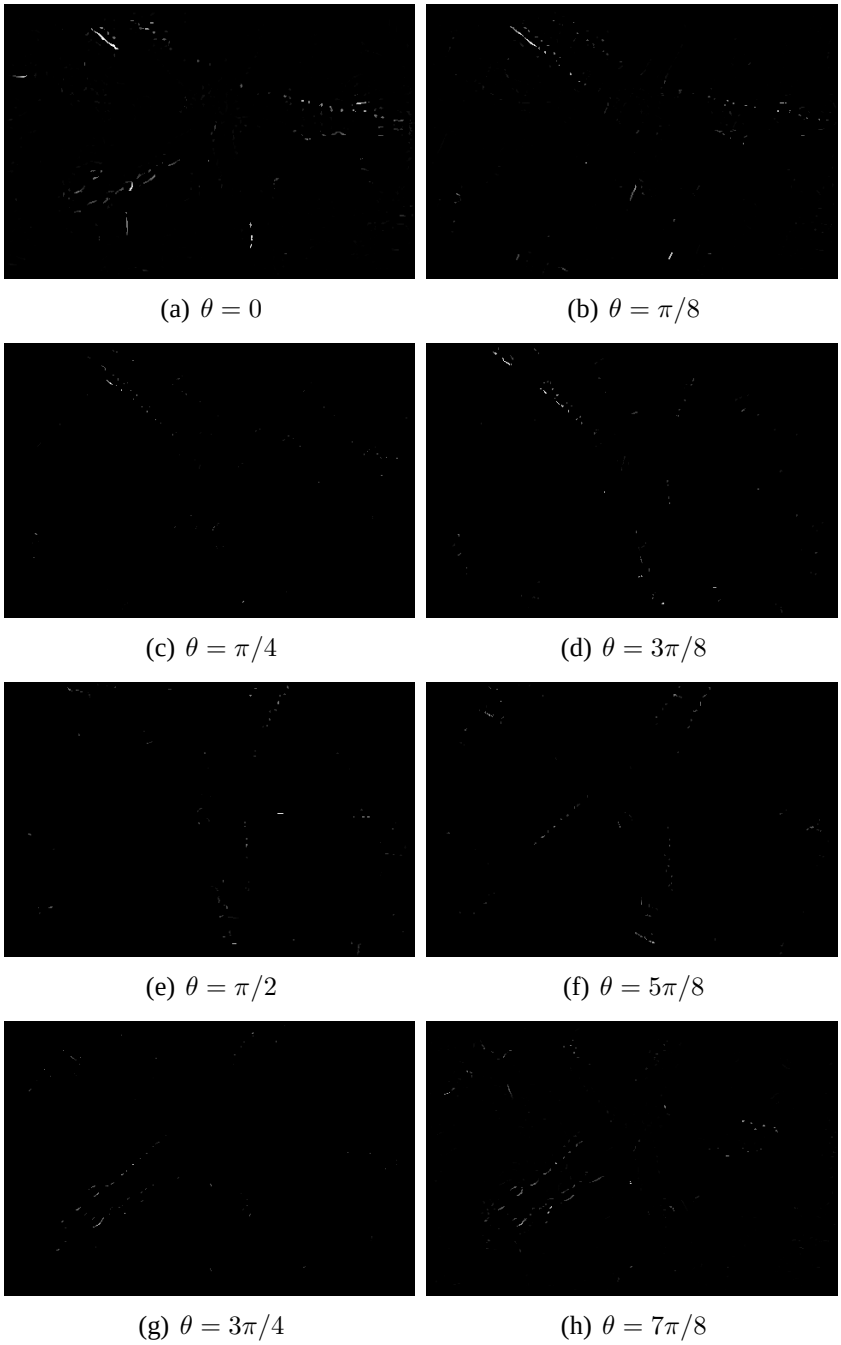


Figure 4.17: Texture channel gradient features, collected at a large scale for the first five orientations. (a): Histogram differences between the left (D_2) and the central (D_1) part of the disk. (b): Histogram differences between the right(D_3) and central (D_1) disk parts. (c): Histogram differences between the left (D_2) and right (D_3) disk parts. Orientation starts at zero degrees (top) and increases clockwise by $\frac{\pi}{8}$ as we move downwards, until $\frac{\pi}{2}$ (bottom row).



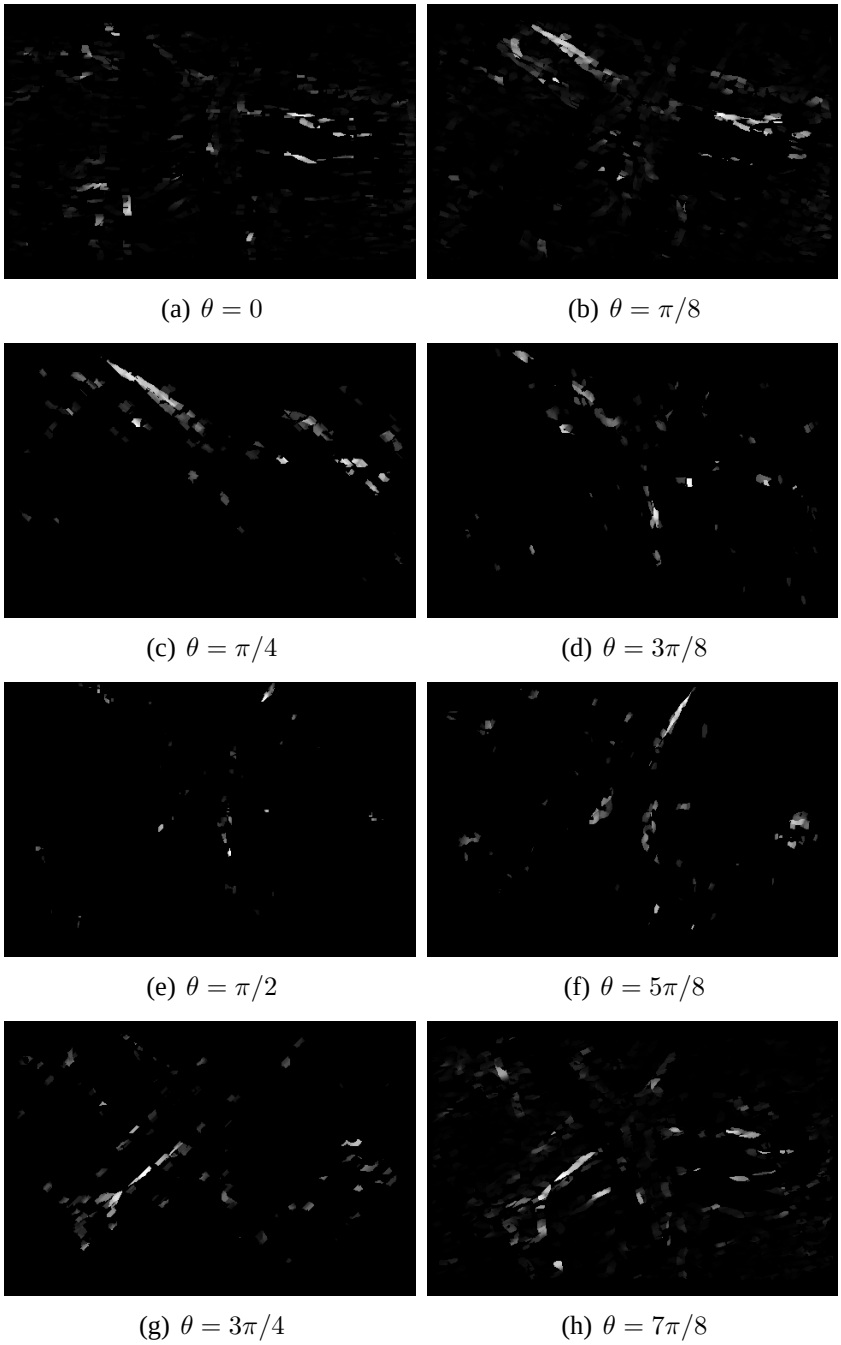
(a) $\theta = 0$ (b) $\theta = \pi/8$

(c) $\theta = \pi/4$ (d) $\theta = 3\pi/8$

(e) $\theta = \pi/2$ (f) $\theta = 5\pi/8$

(g) $\theta = 3\pi/4$ (h) $\theta = 7\pi/8$

Figure 4.18: Boundary validation feature at the smallest scale for all orientations.



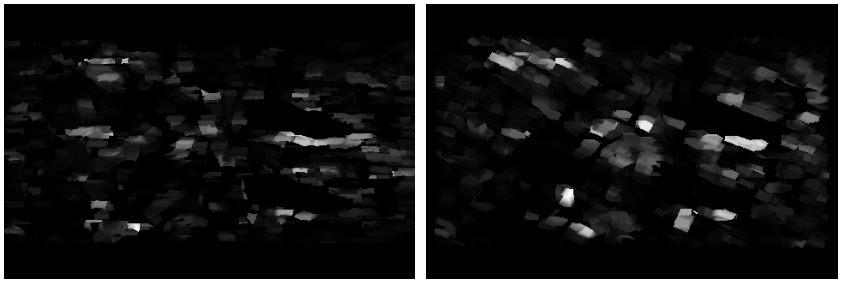
(a) $\theta = 0$ (b) $\theta = \pi/8$

(c) $\theta = \pi/4$ (d) $\theta = 3\pi/8$

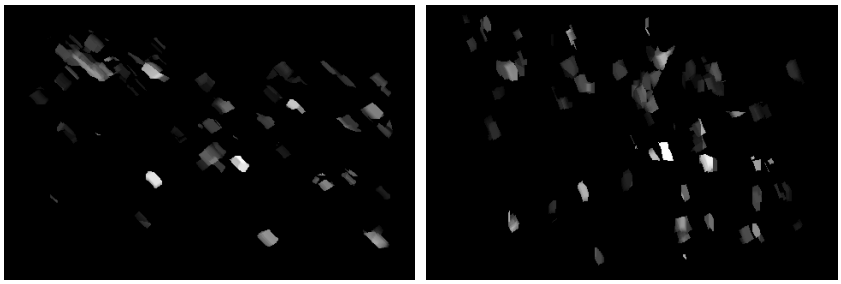
(e) $\theta = \pi/2$ (f) $\theta = 5\pi/8$

(g) $\theta = 3\pi/4$ (h) $\theta = 7\pi/8$

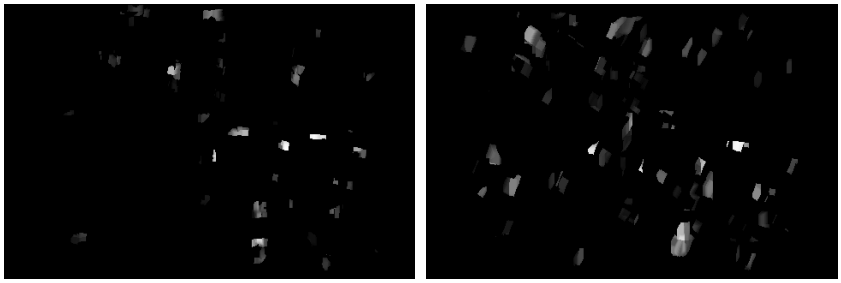
Figure 4.19: Boundary validation feature at a middle scale for all orientations.



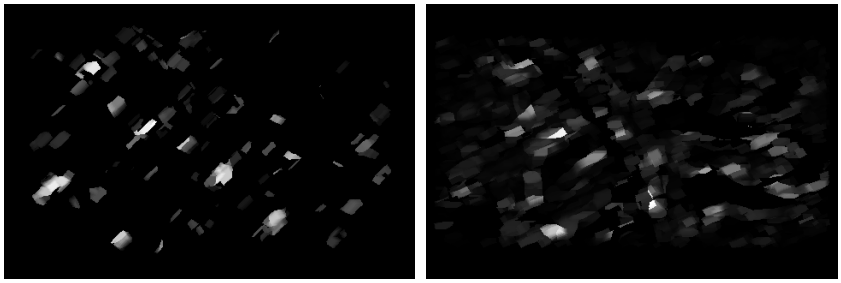
(a) $\theta = 0$ (b) $\theta = \pi/8$



(c) $\theta = \pi/4$ (d) $\theta = 3\pi/8$



(e) $\theta = \pi/2$ (f) $\theta = 5\pi/8$



(g) $\theta = 3\pi/4$ (h) $\theta = 7\pi/8$

Figure 4.20: Boundary validation feature at a large scale for all orientations.

Chapter 5

Learning

In this chapter we provide the theoretical background for *logistic regression* and *multiple instance learning*, which are the classification methods we are going to use in order to train the ridge detector. We also describe the training procedure for both methods.

5.1 Logistic regression

Logistic regression uses a linear model to predict the posterior probability of the occurrence of an event. Despite the simplicity of this model, logistic regression has shown to perform similarly to other, more complex training methods in analogous problems [29]. In the general case of a classification task with K possible classes, the event whose probability we want to measure, is the occurrence of each of the K classes, given the n-vector of independent input variables associated with the output. Since we want to model probabilities, the model is formulated in such a way that the linear functions used, sum to one and remain in [0,1]. In the case of a binary classification problem, i.e. when $K = 2$, there is only a single linear function and the model has the following form:

$$\log \frac{\Pr(G = 1|X = x)}{\Pr(G = 0|X = x)} = w_{10} + w_1^T x \tag{5.1}$$

In the above equation, $G = i$ denotes the occurrence of class i, and x is the vector of input variables. It follows that w_1^T is also a vector and w_{10} is the intercept, which is often included in the parameter vector $w = w_{10}, w_1$, by including the constant term 1 in the input vector. A simple calculation shows that

$$\Pr(G = 1|X = x) = \frac{\exp(w_{10} + w_1^T x)}{1 + \exp(w_{10} + w_1^T x)} \tag{5.2}$$

and of course

$$\Pr(G = 0|X = x) = 1 - \Pr(G = 1|X = x) \tag{5.3}$$

$$= \frac{1}{1 + \exp(w_{10} + w_1^T x)} \tag{5.4}$$

Finally, if we consider that y_i is the 0/1 response to the i-th observation, where $y_i = 1$ when $g_i = 1$, and $y_i = 0$ when $g_i = 2$, the log-likelihood for N observations can be written:

$$L(w) = \sum_{i=1}^{N} \left\{ y_i \log p(x_i; w) + (1 - y_i) \log(1 - p(x_i; w)) \right\} \tag{5.5}$$

$$= \sum_{i=1}^{N} \left\{ y_i w^T x_i - \log(1 + e^{w^T x_i}) \right\} \tag{5.6}$$

In this equation, we assume the simplification mentioned earlier, with the intercept included in the parameter vector and the constant term 1 included in the input vector. The logistic regression model is fit to the data by maximum likelihood. To maximize the log-likelihood, we set its derivatives to zero, getting $p+1$ equations *nonlinear* in w, which are called the *score* equations:

$$\frac{\partial L(w)}{\partial w} = \sum_{i=1}^{N} x_i (y_i - p(x_i; w)) = 0 \tag{5.7}$$

To solve the score equations, we use the Newton-Raphson algorithm, which, given a starting value w^{old}, produces the update w^{new}, using the Hessian matrix of $L(w)$:

$$w^{new} = w^{old} - \left(\nabla^2 L(w) \right)^{-1} \frac{\partial L(w)}{\partial w}, \tag{5.8}$$

where the Hessian matrix is

$$\nabla^2 L(w) = -\sum_{i=1}^{N} x_i x_i^T p(x_i; w)(1 - p(x_i; w)) \tag{5.9}$$

Following [29], for our experiments we set the initial value of w at $w = 0$.

5.2 Multiple instance learning

Multiple instance learning (MIL) is a variation of supervised learning that is very useful in applications in a number of domains, such as computer vision, bioinformatics and text processing. In comparison to logistic regression, MIL offers

the advantage of treating both scale and orientation as latent variables. This way we do not have to search for the feature vectors corresponding to the actual scale and orientation of each training sample; all the feature vectors that are associated to it are used and the actual scale and orientation quantities are inferred by the algorithm. Below we list the basic ideas of MIL, establish the notation that will be used, and compare it to standard supervised learning.

In traditional supervised learning we have a training dataset consisting of input-output pairs. In a classification problem the inputs are the *instance examples*, denoted $\{x_1, x_2 \ldots, x_n\}$, and the outputs are labels that denote the class among a set of K possible classes; for these labels we use the notation $\{y_1, y_2 \ldots, y_n\}$. Instance examples typically lie in $\Re^d$, y_i in $\{0, 1\}$, and the goal is to construct a classifier function $h(X) : \Re^d \rightarrow \{0, 1\}$ that can predict outputs/labels, for novel inputs. In the MIL paradigm, instead of using single input-label pairs, labels are assigned to *sets* of inputs, called *bags*. The training set consists of the set of bags $\{X_1, X_2 \ldots, X_n\}$ and the bag labels $\{Y_1, Y_2 \ldots, Y_n\}$, where $X_i = \{x_{i1}, x_{i2} \ldots, x_{im}\}, x_{ij} \in \Re^d$ and $Y_i \in \{0, 1\}$ for a binary classification problem. A bag is considered *positive* if it contains at least one positive single input; consequently, a bag is considered negative if it does not contain any positive single inputs.

This rule concerning the bag positiveness, can be expressed as follows:

$$Y_i = \begin{cases} 1 & \text{if } \exists j \text{ s.t. } y_{ij} = 1, \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

and the goal of MIL is to ether train an instance classifier $h(x) : \Re^d \rightarrow 0, 1$, or a bag classifier $H(X) : (\Re^d)^m \rightarrow 0, 1$. We note that y_i can also be written this way:

$$Y_i = \max_j y_{ij}, \tag{5.11}$$

so a bag classifier can be easily constructed once the instance classifier is given.

As in the case of logistic regression, the classifier is trained via maximization of the log likelihood function. However, the instance labels are not known during training, so we need to take them into account when we calculate the cost function. The likelihood assigned to a training bag under the multiple instance learning model is:

$$L = \prod_i P_i^{Y_i} (1 - P_i)^{(1-Y_i)} \tag{5.12}$$

where $Y_i \in 0, 1$ is the label of bag i, from a set of n bags and P_i the probability of

a bag being positive. Is it then straightforward to derive that the log-likelihood is

$$\log L = \sum_{i}^{N} \log(P_i^{Y_i}) + \log((1 - P_i)^{(1-Y_i)}) \tag{5.13}$$

$$= \sum_{i}^{N} Y_i \log(P_i) + (1 - Y_i) \log(1 - P_i) \tag{5.14}$$

This equation is similar to 5.6 with the difference that the instance posterior probabilities have been replaced with the posterior probabilities of the bags, whose labels are known during training. The relationship between the bag probabilities and the probabilities of their instances can be naturally defined by the following formula

$$P_i = \max p_{ij}, \tag{5.15}$$

where p_{ij} is the probability of the j-th instance of the i-th bag. Notice that the max operator is not differentiable. To overcome this obstacle, we use a differentiable approximation to the *max*, the *Noisy-OR* (NOR) [48]. The bag probability under the NOR approximation is given by the relationship:

$$P_i = 1 - \prod_{j \in i} (1 - p_{ij}), \tag{5.16}$$

where the probability of an instance being positive is given by the standard logistic function

$$p_{ij} = \frac{1}{1 + e^{-(w_{10} + w_1^T x)}} \tag{5.17}$$

Using the former differentiable definition of 5.16 for the bag probabilities, we can now calculate analytically the log-likelihood gradient with respect to the weight vector w. To begin with, we use the chain law which gives:

$$\frac{\partial \log L(w)}{\partial w} = \sum_{i=1}^{N} \frac{\partial \log L(w)}{\partial p_i} \cdot \frac{\partial p_i}{\partial w} \tag{5.18}$$

We now are going to calculate each term separately for simplicity.

$$\frac{\partial \log L(w)}{\partial p_i} = y_i \frac{1}{p_i} + (y_i - 1) \frac{1}{1 - p_i} = \frac{y_i(1 - p_i) + (y_i - 1)p_i}{p_i(1 - p_i)} \tag{5.19}$$

$$= \frac{y_i - y_i p_i + y_i p_i - p_i}{p_i(1 - p_i)} = \frac{y_i - p_i}{p_i(1 - p_i)} \tag{5.20}$$

$$\frac{\partial p_i}{\partial w} = \frac{\partial \left[1 - \prod_{j \in i} (1 - p_{ij})\right]}{\partial w} = -\frac{\partial \left[\prod_{j \in i} (1 - p_{ij})\right]}{\partial w} \tag{5.21}$$

$$= -\frac{\partial}{\partial w}\left[\frac{e^{w^T(x_{i1}+x_{i2}+\ldots+x_{im})}}{(1+e^{w^T x_{i1}})(1+e^{w^T x_{i2}})\cdots(1+e^{w^T x_{im}})}\right] \tag{5.22}$$

$$= -\frac{\partial}{\partial w}\left[\sigma(w^T x_{i1})\cdots\sigma(w^T x_{im})\cdot e^{w^T(x_{i1}+x_{i2}+\ldots+x_{im})}\right] \tag{5.23}$$

In 5.23, x_{ij} is the feature vector corresponding to the j-th instance of the i-th bag, and $\sigma(x) = (1+e^{-x})^{-1} = e^x(1+e^x)^{-1}$ is the sigmoid function. We also use m to denote the fixed number of instances per bag. Taking advantage of the property of the sigmoid function, according to which $\sigma'(x) = \sigma'(x)(1-\sigma(x))$, equation 5.23 becomes (we have omitted some intermediate calculation steps:

$$\frac{\partial p_i}{\partial w} = -e^{w^T(x_{i1}+\ldots+x_{im})}\sigma(w^T x_{i1})\cdots\sigma(w^T x_{im}) \tag{5.24}$$

$$\cdot\left[\sum_{j=1}^{m} x_{ij} + \sum_{j=1}^{m}\left\{\sigma(w^T x_{ij}) - 1)x_{ij}\right\}\right] \tag{5.25}$$

$$= (1-p_i)\sum_{j=1}^{m}\sigma(w^T x_{ij})x_{ij} \tag{5.26}$$

Finally, substituting 5.26 and 5.20 in 5.18, we get:

$$\frac{\partial \log L(w)}{\partial w} = \frac{y_i - p_i}{p_i}\sum_{j=1}^{m}\sigma(w^T x_{ij})x_{ij} \tag{5.27}$$

This expression is used to minimize the likelihood function and give us the optimal weights w.

5.3 Training

For the training of the classifier via logistic regression, we use a subset of 37 natural images from the BSDS300 dataset for which we have constructed the symmetry axis ground truth data, as described in chapter 3.

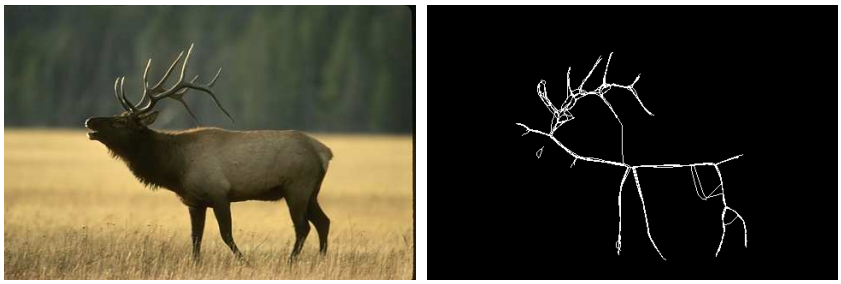
Having in mind that the vast majority of the pixels in each training image are labeled as negatives, and that using all the available pixel/label pairs as training data would increase the computational cost, we pick a subset of the available pixels using a sampling function. The sampling scheme is the following: The total

number of pixels, along with their labelings to be used as training data, is specified as an input parameter for the training function. This total number of samples is divided by the number of the training images to give the number of sampled pixels per image. In our tests, the total number of training samples used (denoted N) was approximately $2 \cdot 10^7$. Training the detector using fewer training samples, in the range of $2 \cdot 10^5 - 5 \cdot 10^5$, and there were not significant variations in the results, neither qualitatively nor in the maximum f-measure attained (see chapter 6). Although the samples are randomly chosen, some constraints are applied. During our first tests, we noticed that boundary pixels are prone to giving a high symmetry response at the smallest scale. This is an unwanted behavior, since boundary pixels cannot by any chance belong to a symmetry axis. For that reason, we emphasize the training against detecting boundary points by choosing half of the boundary points in an image as training samples (labeled as negatives). The labels of the total number of gathered samples that are used in the training do not carry any information as to the orientation or the scale of the symmetry. We therefore have to find a way to select the correct features from the spectrum of collected features at all scales and orientations, corresponding to each sample pixel. For the selection of scale we use a simple approach, using the distance transform on the thresholded boundary map of the image, which is computed by the Berkeley *Pb detector*. Since the features have been collected in discrete scales and not over a continuum, we pick features computed at the scale closest to the distance transform value. For the orientation we use an orientation estimation function using the Hessian of the ground truth image, which gives us the orientation corresponding to the all the positive samples. The default orientation value we use for the pixels that are labeled negative is zero degrees. This way we are able to select appropriate features for training, both in terms of scale and orientation.

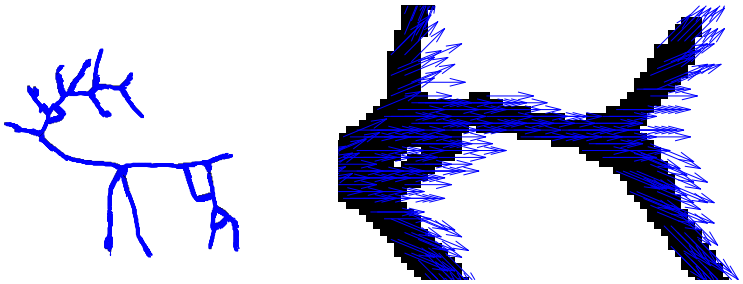
All the features are stacked in a $N \times K$, where K is the length of the feature vector taking into account the constant term. Their labels are also stacked in a single column vector. The training is done using the Newton-Raphson iterative algorithm described in the previous paragraph [16], giving us upon convergence the vector of w_i coefficients.

Training with MIL is done in a similar way to the case of logistic regression, with one basic difference. Recall that in the case of logistic regression, we used estimations for both the scale and the orientation of each sampled pixel, in order to use the correct features for training. In multiple instance learning scale and orientation are treated as hidden variables, so these estimations are not necessary.

Linking the terms used in the previous paragraph to the components of our problem, each pixel represents a bag of features. The instances contained in such a bag are the feature vectors at all orientations and scales. To give a concrete example, in our experiments we used features in eight different orientations and ten different scales for each pixel, which makes for a total of eighty instances



(a) Original image. (b) Ridge map union



(c) Orientation estimation (d) Orientation estimation detail

Figure 5.1: Orientation estimation of the ground truth ridge map union. The orientation in each pixel is visualized by a vector pointing along the respective angle. In this figure, the angles visualized are in the interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$.

contained in each bag. Since all these features are used in the calculation of the instance and bag probabilities, we are presented with the challenge of efficiently manipulating a significant amount of data, both in terms of space and computational cost. In our logistic regression experiments, we used various numbers of training samples, ranging from $2 \cdot 10^5$ to $2 \cdot 10^7$. As explained in section 5.3, there was not substantial change of the testing results when we used fewer samples, hence we settle for $N = 5 \cdot 10^5$, which keeps the training procedure tractable. The goal is once again the maximization of the log likelihood (or equivalently the minimization of the minus log likelihood) of the set of the training bags. For the multiple instance learning paradigm, we choose to perform the log likelihood optimization step using conjugate gradients to compute search directions and on quadratic and cubic polynomial approximations to perform line search.

5.4 Summary

In this chapter we focused on the methods used to train the ridge detector: logistic regression and multiple instance learning. We made a short introduction to the theory lying behind each one of these methods and established the notation for the most important quantities involved. We concluded the chapter describing the training procedure for both cases.

Chapter 6

Testing and evaluation

In this chapter we present the testing results and the methodology used to evaluate the performance of our detector. We begin by briefly describing the testing setup and the post processing performed in the output image. After that we review the *precision-recall* and *F-measure* framework which we use to evaluate the performance of our detector. We present figures with the quantitative results for logistic regression and multiple instance learning, concluding the chapter with a comparison between the two methods and the ridge detection using automated scale selection, presented in [23]. In the end of the chapter we also include some indicative qualitative results obtained from both methods.

6.1 Testing

Testing the detector is performed on a test set of eight images, taken from the BSDS300 test images. Once the detector coefficients have been calculated, testing is applied densely on a test image. Features at all scales and orientations are collected, forming a 4-dimensional space of symmetry response maps. Taking the pixel-wise maximum on this space over all scales and orientations, gives a portrait of the image ridges, like the one in figure 6.1. This can be seen as a rough qualitative depiction of the ridge locations but it has fat responses.

For that reason, we perform non-maximum suppression in order to thin blurry responses and get a connected line marking the symmetry axis. Non-maximum suppression is performed on a 2D input image orthogonally to an angle θ. Since we want to suppress the false responses while keeping the salient ridges connected, it is necessary to perform non-maximum suppression in the correct orientation, which varies from pixel to pixel. To create an orientation map that depicts this change of θ as we move on the image plane, we first compute the maximum of the response probability maps over all scales. That gives us a 3-dimensional

array of maxima-over-scales for each pixel for all orientations. In this array, we search for the orientation that gives the maximum probability for each pixel; this is the orientation which is used to perform non-maximum suppression. Moreover, we zero the weakest responses and keep only the ones that stand out after the non-maximum suppression. This is done by thresholding the suppressed image, at a threshold that is determined through the maximization of the *F-measure*.

6.2 Precision-recall curves

A high quality detector must meet two fundamental criteria. First of all we want the detector to be reliable. That means that if the system marks a pixel as positive, i.e. belonging to a symmetry axis at some scale r and orientation θ, we want this to be true with high probability. Another important property of the detector is completeness, in the sense that if there are points showing symmetry at some scale and orientation, we want our system to be able to detect them and classify them appropriately. These two concepts just described, are termed as precision and recall in the information retrieval community. Formally, *precision* is the fraction of the detections that are true positives, while *recall* is the fraction of true positives that are detected. In probabilistic terms, precision is the probability that the detector's output is valid, and recall is the probability that the ground truth data has been detected. Expressed in mathematical notation, that is:

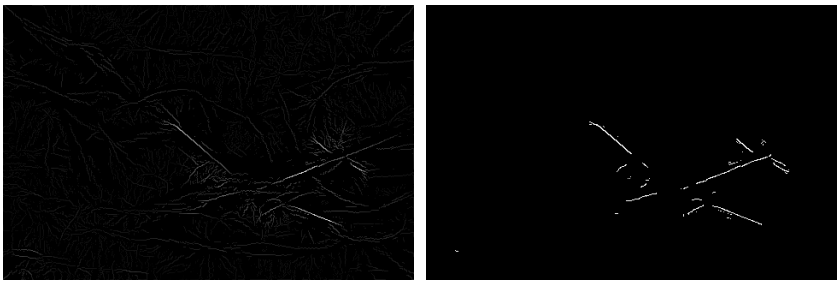
$$\text{Precision} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}} \tag{6.1}$$

$$\text{Recall} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalseNegatives}} \tag{6.2}$$

These two measures are competitive, meaning that if we want a high precision value, we compromise the recall rate of the detector, while achieving high recall rate means the detector will be less precise. To explain this intuitively, a high recall rate means that the detector hasn't missed any positives, but it may have also classified as positives a lot of other pixels, resulting in low precision. High precision on the other hand, means that we are being very selective in our choice of positives; hence most of the positives returned by the detector are indeed true, but a lot of other positives may have not been detected, resulting in low recall.

Precision-recall curves are a common evaluation technique in information retrieval and classification problems and they were first used to evaluate edge detectors by Abdou and Pratt [1]. A similar evaluation technique is the *Receiver Operating Characteristic (ROC)* curves, where the two axes are *fallout* and *recall*. Recall is defined in the same way as above and fallout is the probability

(a) Original image (b) Posterior probability ridge map.



(c) Non-maximum suppression. (d) Filtering and thresholding.

(e) Detected ridges.

Figure 6.1: Processing steps from the initial image to the final ridge map.

that a true negative was falsely labeled as positive. Although both types of curves qualitatively show the same trade-off between misses and false positives, ROC curves are not appropriate for judging the quality of ridge detection. If the image

resolution is increased by a resolution of factor n, the number of pixels grows as n^2. Applying the reasoning found in [29] for ridges this time, lines in $\Re^2$ have a fractal dimension less than 2, so the number of true positives will grow as slow as n, while the number of true negatives will grow as fast as n^2. As a result, the fallout will decline by as much as $1/n$. We do not face the same problem with precision, because it is normalized by the number of positives rather than the number of true negatives.

This trade-off between precision and recall can be captured by the use of an appropriate measure, called the *F-measure*, introduced in [46], which is the weighted harmonic mean of these two measures. Denoting precision as P and recall as R the F-measure is defined as:

$$F = \frac{P \cdot R}{aR + (1-a)P} \tag{6.3}$$

where a lies in [0,1]. In our experiments we use $a = 0.5$, which gives the evenly weighted F-measure.

A precision-recall curve is formed by points representing the F-measure values, each one computed by the P and R values corresponding to the detector's output at a particular threshold. To compute P and R though, we need to be able to determine how many of the pixels detected are true positives, and how many are false positives. Adding to that, the constructed ground truth data consist of 5-10 different symmetry axis maps per image, resulting from the available segmentations for each image in the BSDS300, so we are first going to examine the simpler case of pixel correspondence between a single binary ground truth ridge map and a thresholded detector output. The obvious choice is to consider detected positives as true if they can be corresponded to a pixel in the ground truth image, and declare all unmatched pixels either false positives or misses. This way, however, we don't take into account the localization errors which exist in the ground truth data. Showing intolerance towards these localization errors is bad practice, because qualitatively useful detector results can be rated as low quality and be rejected. To avoid that, we allow correspondence between detected positives and pixels neighboring to the ground truth positives. Details on the correspondence computation can be found in [29].

As multiple humans annotate a single image, the precision-recall curves should be based on all ground truth binary maps available for each image. The approach we follow is to first match the detector result with each ground truth map separately. If a detected positive is matched with at least with one of the binary maps, it is classified as true positive. On the other hand, pixels that correspond to no ground truth ridge map, are declared false positives. The hit rate is averaged over the number of different ridge maps so, in order to achieve perfect recall, all the ground truth data have to correspond to a positive detected by the system.

Finally, since we do not possess a ground truth dataset made by human subjects, we have to find an alternative way of comparing the results of our detector to a "gold standard" F-measure, associated with the annotated set. Recall that for every image of the training dataset, we have a number of segmentations by different human subjects and an equal number of ground truth ridge maps, calculated in the manner described in chapter 3. We evaluate a ground truth F-measure by sequentially treating each one of these ground truth ridge maps as a thresholded detector output and comparing it to the rest. In the end we average the results over the total number of ridge maps. The value that was calculated following this methodology for our ground truth dataset was $F = 0.67$.

6.3 Comparison

6.3.1 Quantitative results

In the figures that follow we compare the F-measure results for the three feature configurations examined in chapter 4, with color channel features used as a single feature or separately, and for the case of gray features (brightness and texture gradients only). Results for logistic regression, multiple instance learning, and ridge detection with automated scale selection are illustrated in the same figure for easier comparison. The number of training samples used for both training methods was $5 \cdot 10^5$ samples.

As we can see in figure 6.2, using multiple instance learning to train the detector, gives the better results compared to logistic regression and the automated scale selection algorithm by Lindeberg (we are going to use the acronym *LASSA* to refer to this algorithm from now on). In terms of the maximum F-measure attained, our MIL detector performs better than LASSA, but stands really close to logistic regression, which also outeperforms LASSA. This result is expected, since the LASSA algorithm is applied on gray scale images, and does not take advantage of the color and texture cues, which play a important role in our approach.

Combining the two color channels into a single feature decreases performance, as we can see in figure 6.3. Both logistic regression and MIL still perform better than ASS, but the difference in the maximum f-measure is now smaller. Moreover, the difference in the maximum f-measure between logistic regression and MIL training has reduced to negligible levels, so practically in this case both methods perform equally well.

Finally, in figure 6.4 we present the results concerning the use of only gray scale features; that is brightness, texture and the boundary validation feature. This is the only case that our detector trained via logistic regression falls behind in comparison to LASSA; the MIL-trained detector still stays ahead however, even

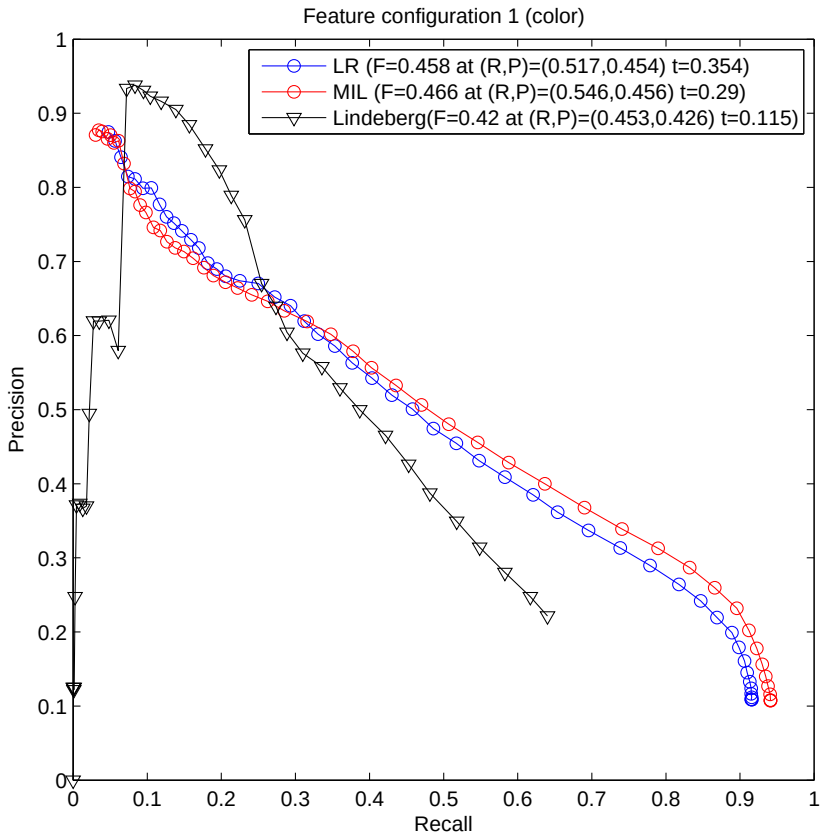


Figure 6.2: Results of evaluation for a detector trained using color channels as separate features. In the legend we note the maximum f-measure achieved, the respective precision and recall values, and the threshold at which it was achieved for each method. Multiple instance learning outperforms all the other methods.

by a small margin. This verifies the importance of the color cue for determining symmetry in the framework we are proposing, and we see that omitting it can lead to great deterioration of the effectiveness of the classifier.

6.3.2 Qualitative results

We observe that there are not significant differences in the detected ridges between the two configurations that use color features. Using the two color channels a^* and b^* as separate features, seems to yield slightly better responses, detecting some parts of the ridges which are not detected when the two channels are used as a single feature (e.g. hand of the person in fig. 6.7). Moreover, both color configurations outperform the case where we use only brightness and texture features.

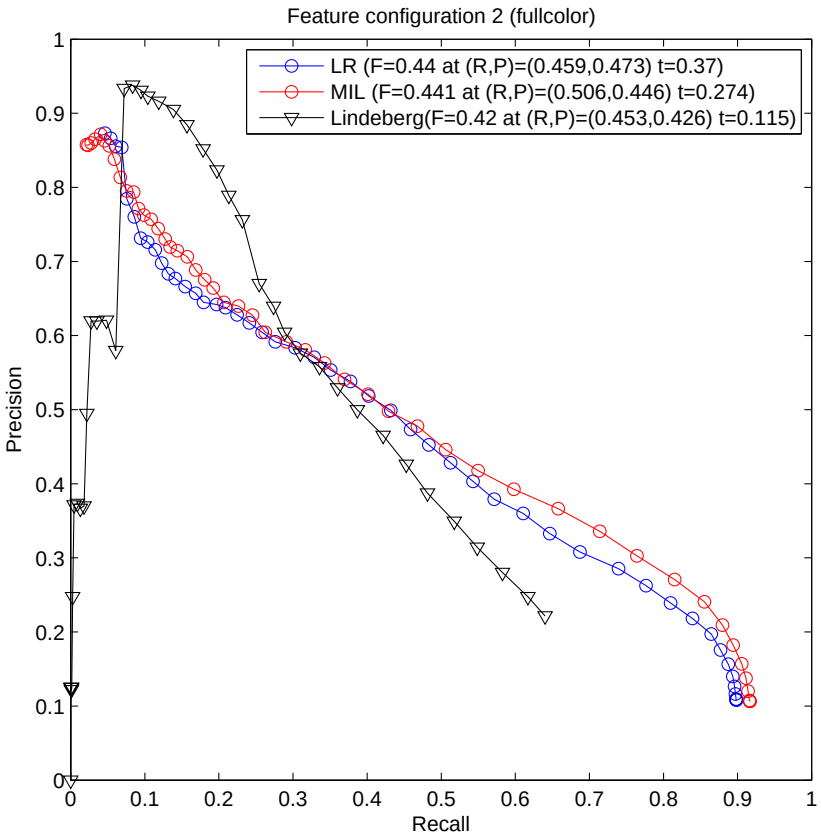


Figure 6.3: Results of evaluation for a detector trained using the sum of the two color channels as single feature. In the legend we note the maximum f-measure achieved, the respective precision and recall values, and the threshold at which it was achieved. Multiple instance learning still performs better than the other methods but at practically the same level as logistic regression.

We can see that the gray scale detector gives a lot of spurious responses; at the same time, it faces difficulty detecting symmetry at larger scale, for example in figure 6.7. In this figure, we see that for color features, the symmetry axis of the person's yellow skirt is successfully located, while the gray scale detector fails to recognize it. The same applies to the torso of the zebra in figure 6.8.

Comparing the results taken from the two training methods, the advantage of using MIL is shown in figure 6.5 where the ridge corresponding to the tail of the plane is more accurately localized for the color feature configurations, whereas in the gray feature case MIL gives fewer background responses. Another example is in figure 6.8 where the ridge corresponding to the torso of the zebra is again detected more accurately. Overall, though, the results by the two methods are

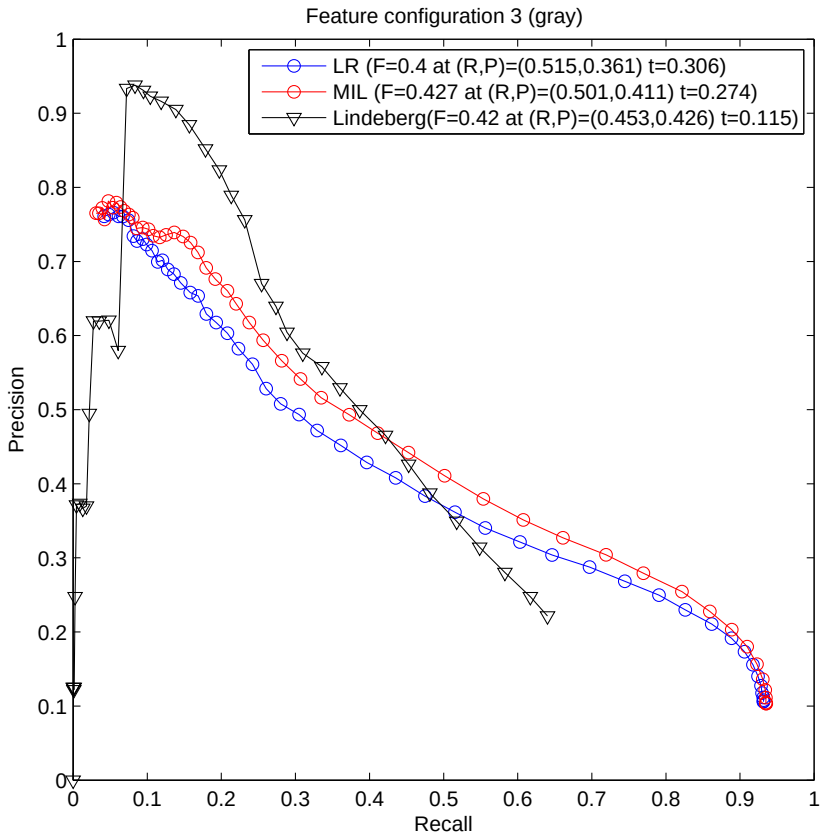


Figure 6.4: Results of evaluation for a detector trained using the sum of the two color channels as single feature. In the legend we note the maximum f-measure achieved, the respective precision and recall values, and the threshold at which it was achieved. Multiple instance learning remains the winner but this time automated scale detection algorithm performs better than our detector trained with logistic regression.

similar.

6.4 Summary

In this chapter we presented the precision-recall framework and we used it to evaluate quantitatively our ridge detector. We presented the results in terms of maximum F-measure for the three different feature configurations used in the training (color, fullcolor and gray scale), and we compared those with the automated scale selection (LASSA) algorithm [23]. Our algorithm manages to outperform ASS,

in most cases, with MIL training providing the best F-measure in all configurations, and an overall maximum at $F = 0.466$ when we treat color channels $a*$ and $b*$ as separate features. An important conclusion we reached is that color is a discriminative cue for symmetry detection, which has a considerable effect on the performance of the detector. Furthermore, we notice that the performance gap between logistic regression and multiple instance learning training methods is not so large, making the former a rather attractive choice, considering its reduced computational cost and complexity. We concluded the chapter presenting some examples of detection results.

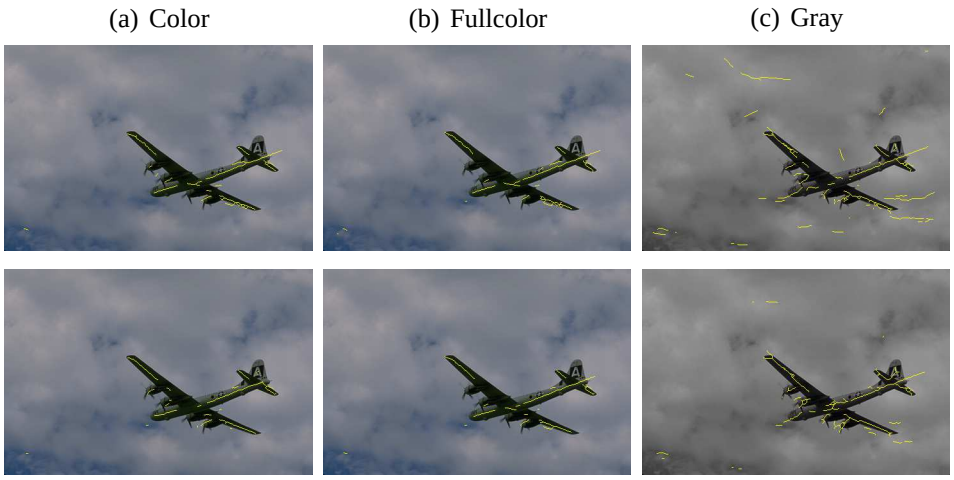


Figure 6.5: Detected ridges for all three feature configurations shown on the initial image ($iid = 3096$ from the BSDS300). Top row shows results for logistic regression and bottom row for MIL.

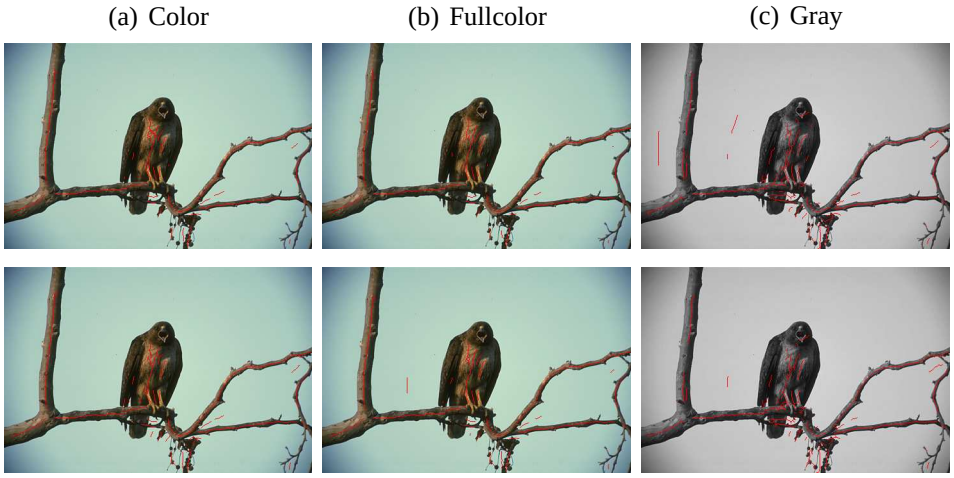


Figure 6.6: Detected ridges for all three feature configurations shown on the initial image ($iid = 42049$ from the BSDS300). Top row shows results for logistic regression and bottom row for MIL.

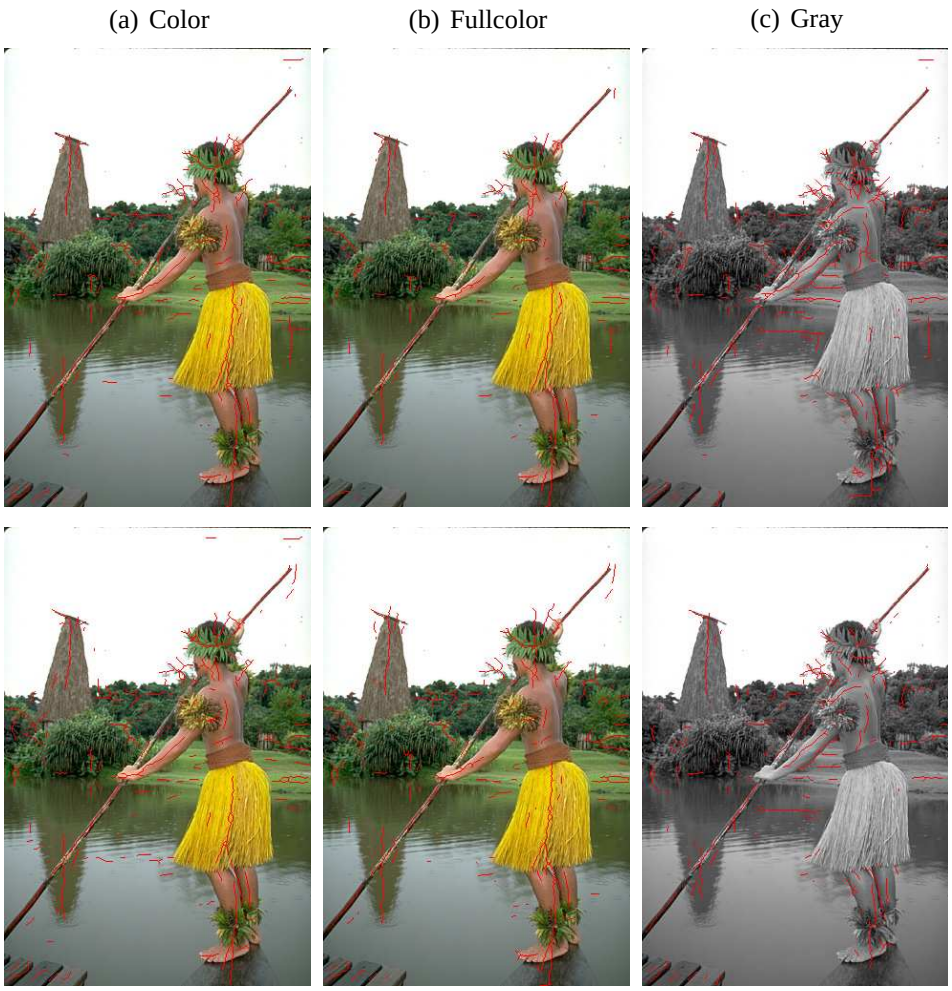


Figure 6.7: Detected ridges for all three feature configurations shown on the initial image ($iid = 101087$ from the BSDS300). Top row shows results for logistic regression and bottom row for MIL.

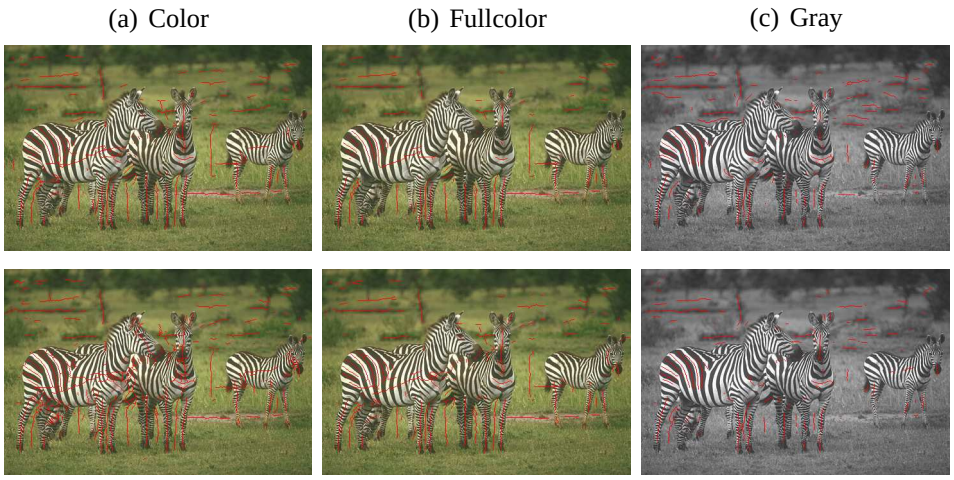


Figure 6.8: Detected ridges for all three feature configurations shown on the initial image ($iid = 253027$ from the BSDS300). Top row shows results for logistic regression and bottom row for MIL.

Chapter 7

Conclusions and future work

7.1 Contribution of the thesis

In this thesis we introduced a novel approach to ridge and symmetry axis detection for elongated image structures, motivated by the increasing popularity of learning methods. More specifically, we developed a ridge detector that seeks symmetry at various scales and orientations, and classifies pixels into ridge/non-ridge classes.

The main contribution of our work is the introduction of a machine learning framework, that uses low-level features to train the detector; the detector then uses the respective feature vectors extracted from a new input image, to decide if a pixel belongs in a ridge. We have used two methods for learning, logistic regression and multiple instance learning, and their results were compared both qualitatively and quantitatively in chapter 6. We saw that multiple instance learning generally performs better by a small margin for the maximum F-measure compared to simple logistic regression. Apart from that, both training methods give better results when compared to Lindeberg's automated scale detection method.

Although our approach seems to perform reasonably well on natural images, using a relatively small training dataset, we believe that its potential would be properly revealed in more targeted applications. Using features tailored to the task and training the detector on a specific subset of objects whose ridges we would like to detect, could significantly improve its efficiency, as the variance in the color and texture cues of the training examples would reduce. An interesting application that could benefit by the use of a ridge detector trained in this fashion is pose estimation, where for example the classifier could be trained to detect the symmetry axes of a person's limbs in an image (hands, legs, torso, head). Our detector could also be trained on aerial images for the extraction of ridges in landscape top views, which is a useful step in a higher level task such as automatic topographic map extraction. Finally, medical imaging is another area where our

method could prove fruitful, e.g. in the detection of ridges in MRI or PET scans, which can sometimes be related to pathological conditions.

7.2 Future work

The results of the method presented in this thesis are encouraging but there are still many future paths that can be explored, and improvements that can be the focus of future research on this work. We list some of the most important below.

First of all, the construction of a human subject annotation dataset for ridges is crucial for the robustness of our system. The ground truth set used in this thesis serves only as a simple starting point in order to test the potential of our method. Thus, building an annotation dataset constructed by real people is one of our first priorities as far as future work is considered.

Secondly, an obvious way to enhance the performance of the detector is to increase the number of features used in training. The most prominent are *SIFT* features [26], which have already been used in learning approaches for boundary detection [18].

Another future direction is using a more sophisticated learning approach, similar to the one employed in [19], where a differentiable approximation of the F-measure serves as the cost to be optimized. More ideas presented in the aforementioned paper could be considered, such as the use of the *anyboost* framework for the classifier training, in conjunction with filtering of the training set via stochastic gradient descent. This is especially useful in the case of multiple instance learning, when the training set is great in volume and selecting an appropriate subset of training instances, without compromising the performance of the classifier.

Inspired by the success of normalized cuts application in image segmentation, demonstrated in the influential paper by Shi and Malik [40], we believe that spectral analysis can also provide us with useful cues for ridge detection. According to the physical interpretation of the generalized eigenvalue problem, given in the former paper, the graph nodes (image pixels) can be considered as physical nodes, and the graph edges as springs connecting each pair of nodes. The affinity matrix, which contains the information of similarity between image nodes, can be adjusted and used as an extra feature in the training step. The boundary information can also be taken into account when we calculate the affinity matrix, in the form of *Intervening Contours*, as explained in [10]. This can be particularly valuable in ridge detection for two reasons: first of all, because the boundary existence at a certain distance and orientation is directly related to a ridge line or a symmetry axis, and second, because sometimes background clutter has similar intensity values to a foreground object, making detection harder.

Finally, we can benefit in terms of time efficiency by porting our code onto the

GPU. Let aside training, which is only performed once in order to determine the beta coefficients of the classifier, the most time-consuming part of the ridge extraction is the collection of the features, which can take several hours in MATLAB if we use a large training set. For example, extracting the features at 10 scales and 8 orientations from 37 images of 321×481 or 481×321 pixels, takes about 5 hours on an Intel i7 processor. We can significantly reduce the time needed, drawing on the fact that feature collection at some scale and orientation is a fully independent process, hence our code can be parallelized. Since the most time-demanding operation in the feature collection algorithm is the convolution of the image with the disk masks, we can also reduce the time needed, applying the *integral image* representation [47].

Bibliography

[1] ABDOU, I., AND PRATT, W. Quantitative design and evaluation of enhancement/thresholding edge detectors. *Proceedings of the IEEE 67*, 5 (1979), 753–763. 59

[2] ADELSON, E. On seeing stuff: the perception of materials by humans and machines. In *Proceedings of the SPIE* (2001), vol. 4299, Citeseer, pp. 1–12. 22

[3] ARBELÁEZ, P., MAIRE, M., FOWLKES, C., AND MALIK, J. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence* (2010), 898–916. 31

[4] BLUM, H. Biological shape and visual science (part i). *Journal of theoretical Biology 38*, 2 (1973), 205–287. 14

[5] BLUM, H., ET AL. A transformation for extracting new descriptors of shape. *Models for the perception of speech and visual form 19*, 5 (1967), 362–380. 14

[6] BRADY, M., AND ASADA, H. Smoothed local symmetries and their implementation. *The International Journal of Robotics Research 3*, 3 (1984), 36. 14, 17

[7] BUSCH, A. A common framework for the extraction of lines and edges. *International Archives of Photogrammetry and Remote Sensing 31* (1996), 88–93. 19

[8] CONNERS, R., AND NG, C. Developing a quantitative model of human preattentive vision. *Systems, Man and Cybernetics, IEEE Transactions on 19*, 6 (1989), 1384–1407. 7

[9] CONWAY, J., BURGIEL, H., AND GOODMAN-STRAUSS, C. *The symmetries of things*. AK Peters Wellesley, MA, 2008. 11

[10] COUR, T., BENEZIT, F., AND SHI, J. Spectral segmentation with multiscale graph decomposition. 71

[11] COXETER, H. *Generators and relations for discrete groups*. 1984. 11

[12] DOLLAR, P., TU, Z., AND BELONGIE, S. Supervised learning of edges and object boundaries. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 2, IEEE, pp. 1964–1971. 7

[13] EBERLY, D., GARDNER, R., MORSE, B., PIZER, S., AND SCHARLACH, C. Ridges for image analysis. *Journal of Mathematical Imaging and Vision 4*, 4 (1994), 353–373. 18

[14] FOGEL, I., AND SAGI, D. Gabor filters as texture discriminator. *Biological cybernetics 61*, 2 (1989), 103–113. 31

[15] FOWLKES, C., MARTIN, D., AND MALIK, J. Local figure–ground cues are valid for natural images. *Journal of Vision 7*, 8 (2007). 22

[16] FRIEDMAN, J., TIBSHIRANI, R., AND HASTIE, T. *The elements of statistical learning: Data mining, inference, and prediction*. Springer-Verlag New York, 2009. 9, 55

[17] HARALICK, R. Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing 22*, 1 (1983), 28–38. 17

[18] KOKKINOS, I. Highly accurate boundary detection and grouping. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, pp. 2520–2527. 71

[19] KOKKINOS, I. Boundary detection using f-measure-, filter-and feature-(f 3) boost. *Computer Vision–ECCV 2010* (2010), 650–663. 71

[20] KONISHI, S., YUILLE, A., COUGHLAN, J., AND ZHU, S. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2003), 57–74. 7

[21] LEVINA, E., AND BICKEL, P. The earth mover's distance is the mallows distance: some insights from statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on* (2001), vol. 2, IEEE, pp. 251–256. 29

[22] LEYTON, M. Symmetry, causality, mind. *The MIT Press, Cambridge, Massachusetts* (1992). 7

[23] LINDEBERG, T. Edge detection and ridge detection with automatic scale selection. In *IJCV* (1998), Kluwer Academic Publishers, pp. 117–154. 19, 58, 65

[24] LIU, Y. Computational symmetry in computer vision and computer graphics. *Foundations and Trends® in Computer Graphics and Vision 5*, 1-2 (2009), 1–195. 12, 14

[25] LÔPEZ, A., LUMBRERAS, F., SERRAT, J., AND VILLANUEVA, J. Evaluation of methods for ridge and valley detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 21*, 4 (1999), 327–335. 18

[26] LOWE, D. Distinctive image features from scale-invariant keypoints. *International journal of computer vision 60*, 2 (2004), 91–110. 71

[27] MALLOWS, C. A note on asymptotic joint normality. *The Annals of Mathematical Statistics* (1972), 508–515. 29

[28] MARAGOS, P. *Image Analysis and Computer Vision*. National Technical University of Athens, 2005. 8

[29] MARTIN, D., FOWLKES, C., AND MALIK, J. Learning to detect natural image boundaries using local brightness, color, and texture cues. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 26*, 5 (2004), 530–549. 7, 10, 21, 28, 31, 32, 50, 51, 61

[30] MARTIN, D., FOWLKES, C., TAL, D., AND MALIK, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on* (2001), vol. 2, pp. 416 –423 vol.2. 21

[31] NABER, G. The geometry of minkowski spacetime. *Washington DC American Geophysical Union Geophysical Monograph Series 1* (1992). 13

[32] O'MARA, D., AND OWENS, R. Measuring bilateral symmetry in digital images. In *TENCON'96. Proceedings. 1996 IEEE TENCON. Digital Signal Processing Applications* (1996), vol. 1, IEEE, pp. 151–156. 19

[33] PALMER, S. *Vision science: Photons to phenomenology*, vol. 1. MIT press Cambridge, MA., 1999. 30

[34] PENROSE, R. The role of aesthetic in pure and applied mathematical research. *The Institute of Mathematics and its Applications 7*, 8 (1974), 10. 13

[35] PIZER, S., EBERLY, D., FRITSCH, D., AND MORSE, B. Zoom-invariant vision of figural shape: The mathematics of cores* 1. *Computer Vision and Image Understanding 69*, 1 (1998), 55–71. 18

[36] PUZICHA, J., BUHMANN, J., RUBNER, Y., AND TOMASI, C. Empirical evaluation of dissimilarity measures for color and texture. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on* (1999), vol. 2, IEEE, pp. 1165–1172. 31

[37] ROSTEN, E., AND DRUMMOND, T. Machine learning for high-speed corner detection. *Computer Vision–ECCV 2006* (2006), 430–443. 7

[38] RUBNER, Y., PUZICHA, J., TOMASI, C., AND BUHMANN, J. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding 84*, 1 (2001), 25–43. 29

[39] SHECHTMAN, D., BLECH, I., GRATIAS, D., AND CAHN, J. Metallic phase with long-range orientational order and no translational symmetry. *Physical Review Letters 53*, 20 (1984), 1951–1953. 13

[40] SHI, J., AND MALIK, J. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*, 8 (2000), 888–905. 71

[41] STEGER, C. An unbiased detector of curvilinear structures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 20*, 2 (1998), 113–125. 19

[42] TELEA, A., SMINCHISESCU, C., AND DICKINSON, S. Optimal inference for hierarchical skeleton abstraction. *Pattern Recognition 4* (2004), 19–22. 21

[43] TELEA, A., AND VAN WIJK, J. An augmented fast marching method for computing skeletons and centerlines. In *Proceedings of the symposium on Data Visualisation 2002* (2002), Eurographics Association, pp. 251–ff. 21

[44] TYLER, C. *Human symmetry perception and its computational analysis.* Lawrence Erlbaum Associates Publishers, 2002. 7

[45] VAN EEDE, M., MACRINI, D., TELEA, A., SMINCHISESCU, C., AND DICKINSON, S. Canonical skeletons for shape matching. — (2006), 64–69. 17, 21

[46] VAN RIJSBERGEN, C. Information retrieval, chapter 7. *Butterworths, London 2* (1979), 111–143. 61

[47] VIOLA, P., AND JONES, M. Rapid object detection using a boosted cascade of simple features. 72

[48] VIOLA, P., PLATT, J., AND ZHANG, C. Multiple instance boosting for object detection. *Advances in neural information processing systems 18* (2006), 1417. 53

[49] WATSON, J., AND CRICK, F. A structure for deoxyribose nucleic acid. *A century of Nature: twenty-one discoveries that changed science and the world* (2003), 82. 13

[50] WEYL, H. *Symmetry*. Princeton Univ Pr, 1983. 11